
MAGAZINE
MSX
AÑO II
Núm. 14
Junio
1986
300 Ptas.
Memoria de vídeo:
Los sprites
TYPE MISMACHT
SYNTAX ERROR
OUT OF DATA
Código máquina:
Los registros dobles
Aplicaciones matemáticas:
Interpolación
Libros, programas, software
CONTROLE SUS ERRORES DE PROGRAMACION

DIRECTOR:
Juan Arencibia.
COORDINADOR EDITORIAL:
J. Ignacio Rey.
COLABORADORES:
Octavio López, Angel Zarazaga, Teresa Aranda, Ricardo García.
DISEÑO:
Benito Gil
Editada por:
PUBLINFORMATICA, S.A.
C/ Bravo Murillo, 377 - 5.º A
Tel.: 733 74 13
28020 Madrid.
Telex 48877 OPZXE
PRESIDENTE:
Fernando Bolín.
DIRECTOR EDITORIAL REVISTAS DE USUARIOS:
Juan Arencibia.
DIRECTOR DE VENTAS:
Antonio González.
JEFE DE PRODUCCIÓN:
Miguel Onieva.
SERVICIO AL CLIENTE:
Julia González.
Tel.: 733 79 69
DIRECCION, REDACCION Y ADMINISTRACIÓN:
C/ Bravo Murillo, 377 - 5.º A.
Tel.: 733 74 13
28020 Madrid.
PUBLICIDAD EN MADRID:
Emilio García.
PUBLICIDAD EN BARCELONA:
Lidia Cendros.
C/ Pelayo, 12.
Tel.: (93) 301 47 00 Ext. 27-28.
08001 Barcelona.
Depósito Legal: M. 16.755-1985
Imprime: G. Velasco, S. A.
C/ Torrelara, 8. 28016 Madrid.
Distribuye:
S.G.E.L. Avda. Valdelaparra, s/n.
Alcobendas (Madrid).
DISTRIBUIDORES:
VENEZUELA: SIPAM, S.A.
Avda. República Dominicana, 541
ARGENTINA: DISTRIBUIDORA INTERCONTINENTAL
BUENOS AIRES.

El P.V.P. para Ceuta, Melilla y Canarias, incluido servicio aéreo será de 300 ptas. sin I.V.A.

SUSCRIPCIONES:
Rogamos dirija toda la correspondencia relacionada con suscripciones a:
MSX
EDISA. Tel. 415 97 12
C/López de Hoyos, 141-5.º
28002 MADRID
(Para todos los pagos reseñar solamente MSX)
Para la compra de ejemplares atrasados diríjanse a la propia editorial
MSX
C/Bravo Murillo, 377-5.º A
Tel. 733 74 13 28020 MADRID

Si deseas colaborar en MSX remite tus artículos o programas a Bravo Murillo 377, 5.º A. 28020 Madrid. Los programas deberán estar grabados en cassette y los artículos mecanografiados.
A efectos de remuneración, se analiza cada colaboración aisladamente, estudiando su complejidad y calidad.



Con el verano en puertas, pocas son las novedades que aparecen en el marcado. Sólo la aparición de algunos programas y ordenadores más o menos preparados, rompen con la expectativa de la época estival, donde el usuario apreciará un leve estancamiento por parte de los fabricantes.

Hasta la fecha, el lanzamiento de la 2.ª generación por parte de Philips y Sony, mantiene un poco la expectativa del resultado incierto.

Ahora bien, del Reino Unido soplan otros vientos que nos traen varias innovaciones que afectarán de forma indirecta al estándar MSX. Se debe a la aparición de ordenadores personales, que inicialmente estaban configurados para trabajar con discos de 5.25 pulgadas, y que ahora lo hacen con los de 3.5 pulgadas. El Philips: YES, IBM y Toshiba, tienen modelos que permiten trabajar con estos formatos (además estos dos últimos son portátiles). A primera vista esto no quiere decir nada, pero como quiera que la 2.ª generación viene con ese aire de semiprofesionalidad, podremos estar presenciando una nueva etapa, donde el usuario podrá trabajar en la oficina con su ordenador y utilizar el disco en casa para continuar con la labor en un MSX. Este caso pronto dejará de ser una utopía.

Por otro lado, este mes tocamos un tema muy interesante, sobre todo para aquel que gusta de escribir sus propios programas; los errores de programación. Efectivamente, aunque no sea nada nuevo, los errores de programación son una constante en los programas. Sin embargo, una de las características del BASIC MSX es que posee un lenguaje de programación que permite detectarlos y, de alguna forma, controlarlos. De esta manera, el usuario se ve protegido contra estos errores, que en la mayoría de los casos son inesperados.

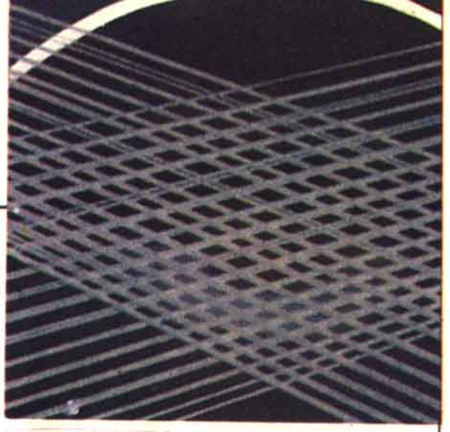
Por otro lado, la memoria de vídeo vuelve a nuestras páginas con la explicación de los sprites. Llega en un buen momento, justo para aprovechar las vacaciones y experimentar todo lo expuesto a lo largo de los meses anteriores.

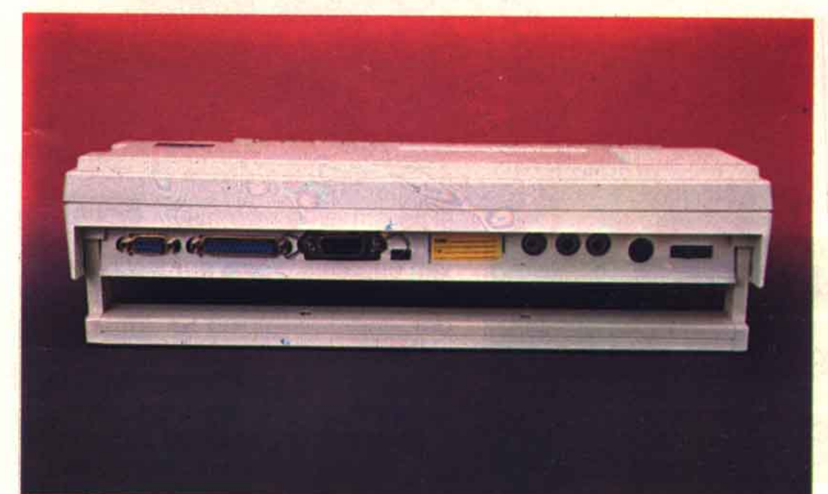


SUMARIO











NOTICIAS



Nuevos juegos para MSX

El mercado del software está en alza. Ya se empiezan a ver juegos nuevos, en teoria, para ordenadores MSX. PROEINSA, que distribuye los juegos de Activision, ha lanzado ocho títulos que van desde Ghostbusters hasta el conocido H.E.R.O. Todos se han comercializado en otro tipo de ordenadores, tales como el Commodore 64 y el Spectrum y han tenido bastante éxito. Hemos de destacar que dichos juegos han ganado en rapidez y en colorido, hecho que constatamos al probar todos y cada uno de ellos (los cuales irán apareciendo en nuestra crítica de software en meses sucesivos).

Este hecho no hace más que afianzar al estándar en el complejo mercado español, que está deseoso de ver aparecer programas interesantes, ya sean juegos o aplicaciones.

También estamos pendientes de ver los primeros paquetes de aplicaciones para los ordenadores de la 2.ª Generación, que por el momento van encaminados a aprovechar las capacidades gráficas del ordenador.

De cualquier manera, el verano se presenta muy caliente en el tema de software.

De la mano de SERMA, aparecerán en breve varios juegos muy interesantes y de características muy peculiares. El primero de ellos es el Juego del Cometa, apoyado en el hecho científico del cometa Halley. De él se dice que desde hace tiempo varios científicos se muestran a favor de la siguiente teoria. La atmósfera que rodea al cometa Halley es capaz de mantener una forma de vida muy básica, de bacterias...

El juego en cuestión está listo para salir al mercado (lo hará en fechas próximas), ahora que el cometa se marcha de nuestro planeta.

De apariencia similar a los típicos juegos galácticos, el Juego del Cometa es distinto en cuanto a la toma de decisiones y al objetivo del juego.

No es la única novedad que se presenta en el mercado. Hace un mes Idealogic lanzaba sus primeros programas de aplicación en cartucho; IDEA TEXT e IDEA BASE. Ambos muy buenos y que permitirán aprovechar hasta el máximo las cualidades de cualquier ordenador.

DINSA: nueva línea de ataque

Desarrollo Informático, S.A. es una compañía perteneciente al grupo de empresas TUDOR, cuyo objetivo es la comercialización de ordenadores personales y periféricos. Principalmente dedicado a ordenadores como IBM y APPLE, DINSA no descarta trabajar con otro tipo de ordenadores y a tal fin a abierto una cadena de tiendas (COMPUWORLD) destinada a la venta de ordenadores domésticos, así como ordenadores de segunda mano.

Actualmente, existen 11 delegaciones de DINSA repartidas a lo largo y ancho de nuestra geografía, emplazadas en Madrid, Bilbao, Zaragoza, Valencia, Alicante, Valladolid, Barcelona, Sevilla, San Sebastián, Granada y Castellón, y pronto en otras provincias.

Más libros Anaya

Anaya Multimedia presenta, en la Feria del Libro, una serie de libros interesantes que abarcan el más amplio espectro. En mayo aparecieron dos libros de Tim Hartnell titulados: «Sistemas Expertos, Introducción al diseño y aplicaciones» y «Simulaciones. Replica la realidad con tu ordenador» a un precio de 2.120 y 1.643 ptas. respectivamente. Tim Hartnell es conocido por sus diversos trabajos y publicaciones.

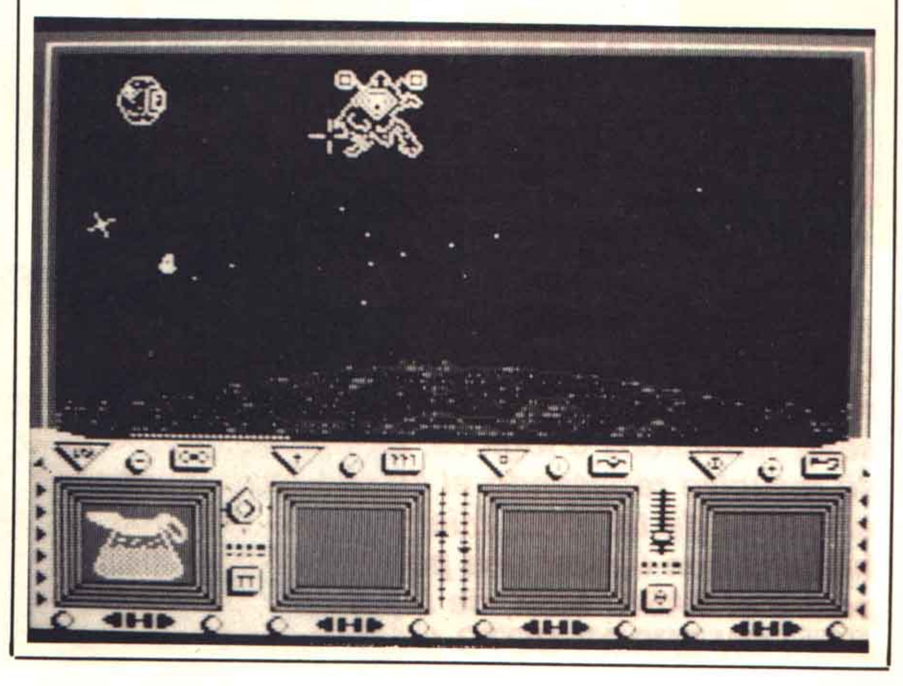
Para la Feria del Libro se presentarán tres libros: «Satélites y Estaciones Espaciales», «Navegación» y «Robots». El primero a 477 ptas. y los dos siguientes a 1.000 ptas.

Para junio, veremos dos obras completas e interesante aunque a un nivel más elevado. No podían faltar aquellos dedicados a lenguajes poco conocidos, tal es el caso de «LISP. El lenguaje de la Inteligencia Artificial» y «PROLOG. Programación y aplicaciones en Inteligencia Artificial» ambos de A. A. Berk y al mismo precio, 1.378 ptas.

En septiembre y octubre, veremos una serie de libros dedicados a los IBM PC, preparados para facilitar la tarea de manejar algunos de los paquetes del PC. Este es el caso de «Técnicas avanzadas en Framework» de Alan Simpson, «Programación en Pascal y Turbo Pascal» de Rodnay Zaks y Trucos y Recursos en Symphony» de D. L. Feldman.

Expreso de oriente

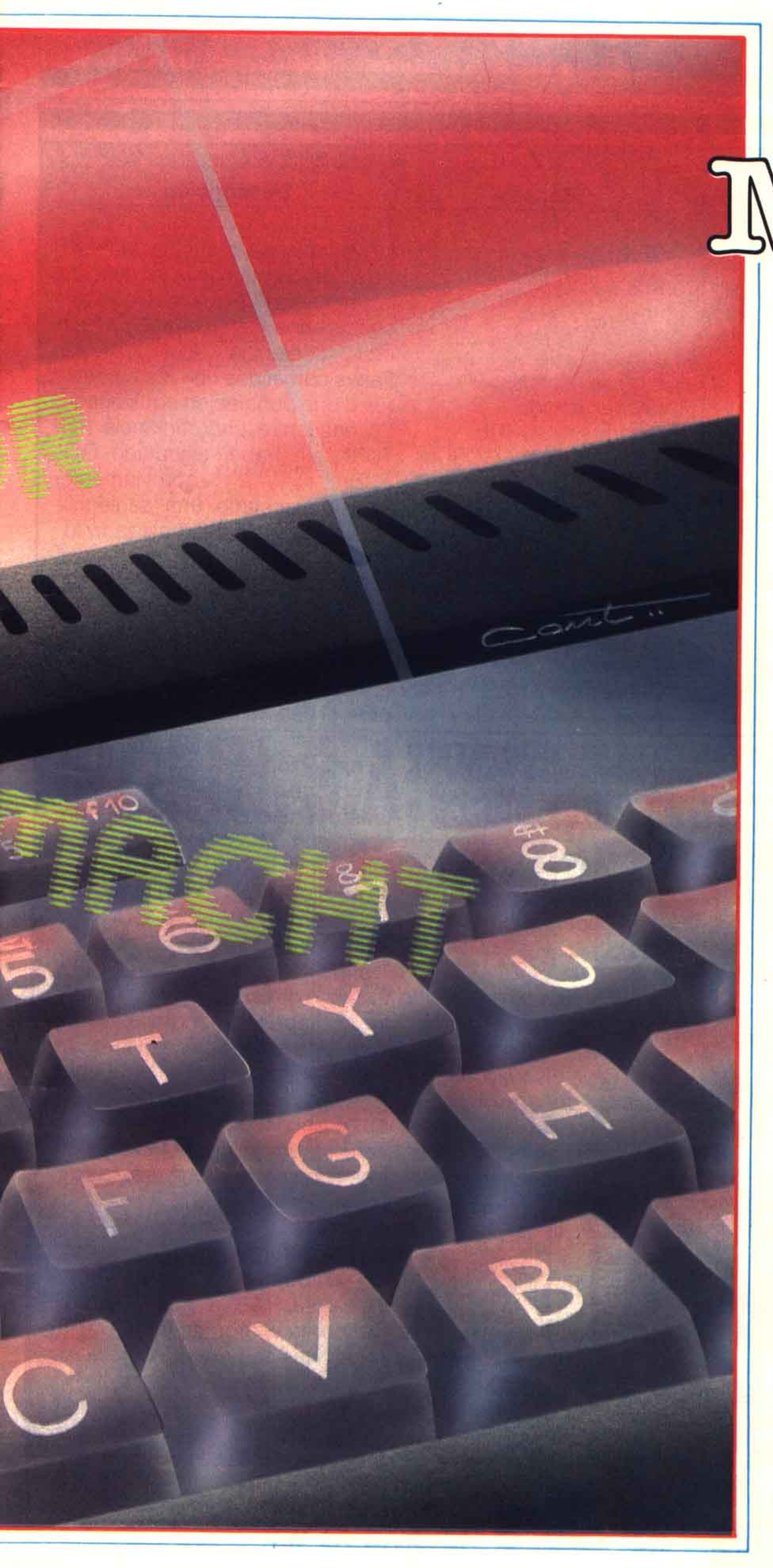
Aunque no sea de Oriente, desde Holanda nos han llegado varias noticias concernientes al estándar. Por ejemplo, la disputa sobre MSX entre Microsoft y ASCII ha tenido un final feliz, lo que significa que ambas empresas se separan pero sin dejar colgando a MSX. Esperemos que ASCII apoye decididamente al MSX, como así lo aseguró Kay Nishi. Mientras tanto, el debate sobre los discos de 3.5 pulgadas cuenta con un nuevo aliado. IBM, interesado en dicho formato, comercializa (aunque no en nuestro país) la versión 3.2 del popular DOS. Este paso, tan esperado y deseado, muestra el interés que tiene esta empresa por este creciente soporte de almacenamiento. De cualquier manera, IBM no se lanzará con este formato durante algún tiempo, principalmente debido a que el número de unidades de éste aún es mínimo. En Europa, la cantidad de unidades de disco no sobrepasa el 20% ó 25% de los poseedores de ordenadores domésticos, lo que quiere decir que el mercado es ínfimo para asegurar una penetración mínima. Pero como es lógico, el tiempo será un factor a tener en cuenta.



Controle los errores de programación



Uno de los principales problemas que plantea la realización de programas son los errores, ya sean sintácticos o lógicos. En muchos casos esto se resuelve repasando el listado hasta encontrar el motivo de tal error, mientras que en otros bastaría con utilizar una potente instrucción que posee el BASIC MSX, de la que poco o nada se sabe: ON ERROR GOTO (número de línea).



Muchos usuarios, expertos o no, utilizan con poca frecuencia una característica típica del *BASIC* de *Microsoft,* que permite detectar y controlar los errores. Otros a lo mejor ni siquiera saben que poseen una instrumentación que ayuda a realizar esta operación y en el peor de los casos, saben que existe una forma de detectar errores pero no cómo aplicarlo a los programas. De manera que, para todos aquellos que no la han utilizado o no se atrevieron con ella, hemos elaborado este artículo que esperemos resulte de gran utilidad, sobre todo para aquellos que gusten de realizar sus propios programas.

La teoría es relativamente simple, mientras que la práctica se adquiere haciendo programas y aplicando esta potente utilidad.

En realidad, la instrucción *ON ERROR GOTO* (número de línea), es una característica que posee el *BASIC* de *Microsoft,* que permite controlar todo tipo de errores, desde los más simples hasta los complicados. Este comando hace que podamos detectar, y actuar en consecuencia, los errores que aparezcan en un programa durante su ejecución, evitando situaciones engorrosas como cuando se bloquea el ordenador. ¿En qué situaciones se pueden detectar errores? En cualquier momento, durante la ejecución de un programa, puede saltar la sorpresa y comprobar, como una instrucción, aparentemente correcta resulta ilógica (aunque este tipo de error es más difícil y no existen trampas para detectarlo).

El *BASIC* de *Microsoft,* puede reconocer un total de 255 condiciones erróneas (en el caso particular de MSX), cada una con su código, que veremos más adelante.

Pero antes veamos qué hace el

ordenador. Cuando un programa se ejecuta y el intérprete detecta un posible error, se asigna el número de código de error a la función reservada denominada *ERR,* mientras que el número de la línea donde se ha detectado dicho error lo podremos encontrar en otra función reservada denominada *ERL,* finalizando así la ejecución del programa y mostrando en la pantalla el mensaje de error correspondiente, con el número de línea y el tipo. Si el error es de lógica, entonces nos viene bien que el ordenador se «cuelgue» ya que los programas se han de repasar concienzudamente antes de hacerlos, puesto que este tipo de error es bastante difícil de detectar y sin embargo, muy fácil de corregir. Ahora bien, en muchos casos, las condiciones de error se generan porque la información introducida es incorrecta o no es lo que el programa esperaba.

SYNTAX ERROR
TYPE MISMACHT
OUT OF DATA

Las variables ERR y ERL son determinantes a la hora de detectar y corregir errores.

Por ejemplo, en un buen programa de aplicación, se espera que estos errores no sólo se puedan detectar desde el programa, sino que se puedan realizar las acciones oportunas para evitarlos, sin mostrar mensajes de error alguno. Con el comando *ON ERROR,* el ordenador saltará a la

Código	Mensaje	Descripción
1	NEXT sin FOR *(NEXT without FOR)*	La variable de la sentencia NEXT no se empareja con la variable de la sentencia FOR anterior, o se encuentra una instrucción NEXT sin una sentencia FOR previa.
2	Error de sintaxis *(Syntax error)*	Faltas cometidas con los caracteres o una puntuación equivocada.
3	Return sin GOSUB *(Return without GOSUB)*	Se encuentra una sentencia RETURN sin haber ejecutado GOSUB previamente.
4	Fin de datos *(Out of DATA)*	Se ha ejecutado una sentencia READ cuando se han utilizado todos los datos de las sentencias DATA de donde leer (READ).
5	Llamada ilegal a una función *(Illegal function call)*	Se pasa un parámetro ilegal a una función matemática o de tipo de cadena de carácteres.
6	Desbordamiento *(Overflow)*	El número es demasiado grande para el tipo de variable, función o sentencia que lo sustenta.
7	Desbordamiento de memoria *(Out of Memory)*	La ejecución se sale de la memoria porque el programa es muy extenso; tiene demasiados FOR, demasiados GOSUB, demasiadas funciones o variables.
8	Número de línea no definido *(Undefined line number)*	No existe el número de línea referido en un GOTO, un GOSUB, una sentencia IF/THEN/ELSE o un comando DELETE.
9	Indice fuera de rango *(Subscript out of range)*	El índice de un elemento de la matriz sobrepasa el rango declarado en DIM, o el índice no es correcto.
10	Matriz ya diseminada *(Redimensioned Array)*	Dimensionas una matriz ya existente utilizando DIM.
11	División por cero *(Division by zero)*	Se ha encontrado una división por cero en una expresión.
12	Ilegal en modo directo *(Ilegal direct)*	Una sentencia que no está permitida en modo directo.
13	Error de tipos *(Type mismacht)*	Se asigna un número a una variable de tipo cadena de carácteres o viceversa.
14	Sobrepasa el espacio de una cadena de carácteres *(Out of string space)*	Desborda el área de trabajo asignado a las cadenas de carácteres en la RAM del sistema.
15	Cadena de carácteres demasiado larga *(String too long)*	Se ha creado una cadena de carácteres de más de 255 carácteres de longitud.

Código	Mensaje	Descripción
16	Fórmula de cadena de caracteres muy compleja *(String formula too complex)*	Expresión de cadena de caracteres muy compleja para que la ejecute el ordenador.
17	Continuación imposible *(Can't continue)*	Intento de continuación en un programa que se ha parado por un error, o que acaba de ser editado o ha finalizado su ejecución en END.
18	Función de usuario no definida *(Undesfined user function)*	Llamada a una función FN sin haberla definido antes con la sentencia DEF FN.
19	Error del dispositivo de E/S *(Device I/O Error)*	Error en la entrada/salida del cassette, impresora, etc.
20	Error de verificación *(Verify error)*	Es un error fatal sin posible remedio. CLOAD? ha detectado un error en la verificación del cassette.
21	No tiene RESUME *(No RE-*	Una rutina de tratamiento de erro-

línea especificada y efectuará la acción oportuna. Para volver al programa principal hace falta utilizar cualquiera de las instrucciones siguientes:
RESUME, RESUME NEXT, RESUME (número de línea).

Códigos de error

Como ya indicábamos anteriormente, el *BASIC* de *Microsoft* describe unos 47 códigos de error. Sin embargo, MSX permite detectar hasta 255 códigos de error, aunque hay algunos que no se utilizan. Los códigos 1 a 25 están dedicados a los errores de sintaxis, como por ejemplo; *NEXT* without *FOR (NEXT* sin *FOR), Syntax error*

(Error sintáctico), etc. Los códigos de error 50 a 67 ambos inclusive, hacen referencia a los errores generados en las operaciones de entrada/salida en cinta o en disco. Es aquí donde la rutina ON ERROR GOTO muestra su verdadera potencia.

Luego, existen una serie de códigos de error, 26 a 49 y 60 a 255, que no se usan para nada y están

SYNTAX ERROR
TYPE MISMACHT
OUT OF DATA

Los códigos de errores, junto con las rutinas adecuadas, permiten que un programa tenga mejor presentación y mayores posibilidades.

reservados para futuras ampliaciones del sistema.

De cualquier manera, la mejor forma de comprobar y aprender el funcionamiento de esta instrucción, es con un caso práctico. Para ello, partiremos del fragmento de un programa (listado 1), que está escrito en pseudo *BASIC* con errores hechos a propósito. Insistimos en que se trata de un supuesto programa de aplicación (si lo introducimos en el ordenador y se ejecuta nos encontraremos con muchos errores).

El fragmento del programa hace referencia a una operación frecuentemente realizada en programas de aplicación y de gestión de ficheros. Los datos se leen de un fichero y como resultado de ello (y en función de los datos) necesita-

Código	Mensaje	Descripción
	SUME)	res no tiene sentencia RESUME cuando el ordenador la está esperando.
22	RESUME sin error *(RESUME without error)*	Encuentra una sentencia RESUME fuera de una rutina de error.
23	Error sin mensaje *(Unprintable error)*	Se ha encontrado un error que no tiene mensaje.
24	Falta un operando *(Missing operand)*	Una expresión contiene un operador sin un operando que la siga.
25	Desbordamiento de una línea *(Line buffer overflow)*	Una línea tiene demasiados carácteres para entrar en el buffer.
26-49	Error sin mensaje *(Unprintable error)*	Reservado para futuras extensiones.
50	Desbordamiento del campo *(Field Overflow)*	Una sentencia de campo (FIELD) abarca más bytes de los especificados por la sentencia OPEN indicando la longitud del registro de un fichero de acceso aleatorio, o estamos al final del buffer mientras hacemos una E/S secuenciál (PRINT = INPUT =) a un fichero de acceso aleatorio.
51	Error interno *(Internal error)*	Mal funcionamiento de la máquina.
52	Error en el número de fichero *(Bad file number)*	Una sentencia o comando hace referencia a un fichero inexistente o fuera del rango indicado por MAXFILES.
53	Fichero no encontrado *(File not found)*	Una sentencia LOAD u OPEN hace referencia a un fichero inexistente.
54	Fichero ya abierto *(File already open)*	Intento de abrir un fichero que ya estaba abierto.
55	Intento de lectura después del fin de fichero *(Input past end)*	Se ejecuta una sentencia INPUT después de que se hayan acabado todos los datos del fichero. Emplea EOF para prevenir este error.
56	Error en el nombre del fichero *(Bad file name)*	Das un nombre ilegal a un fichero con las sentencias LOAD, SAVE u otras de E/S.
57	Sentencia directa dentro de un fichero *(Direct statement in file)*	Se ha encontrado una instrucción directa mientras se está haciendo una operación de carga en memoria. La carga se ha acabado.
58	E/S secuencial *(Secuential I/O only)*	Se pretende hacer un acceso aleatorio en un fichero secuencial.
59	Fichero no abierto *(File not OPEN)*	Se hace referencia a un fichero que no está abierto.
60-255	Error sin mensaje *(Unprintable error)*	No tienen códigos de error. Los puede definir el usuario.

```
Figura 1: Problemas potenciales del programa.

LINEAS      ERR     RAZON PROBABLE
1010        56      Nombre del fichero equivocado
1510
1020        66      Disco lleno
1040
1530        67      Directorio lleno
1040        61      Nombre del fichero equivocado ó en
diferente formato
1100        65      "VIEJO.BAK" existe
1010        55      Se ejecuta una lectrua después del fin del
fichero
1510
1020        60      Formato incorrecto
1510
```

Fig. 1.

```
130 VIEJO.FICH$=FICHERO.DOS$: NAME "fichero.dos$" AS "fichero.dos$"
150 VIEJO.FICH$=FICHERO.TRES$: NAME "fichero.tres$" AS "fichero.tres$"
170 VIEJO.FICH$=FICHERO.CUATRO$: NAME "fichero.cuatro$" AS "fichero.cuatro$"
```

Fig. 2.

```
2010 IF ERR=60 THEN PRINT VIEJO.FICH$;"esta en formato erróneo";:RESUME <ERL-10>
```

Fig. 3.

```
2010 IF ERR=60 THEN PRINT VIEJO.FICH$;"esta en formato erróneo";:RESUME 2020
2020 REM Rutina para err=60
2030 IF ERL=130 THEN RESUME 120: IF ERL=150 THEN RESUME 140: IF ERL=170 THEN RES
UME 160:
```

Fig. 4.

```
2010 IF ERR=64 THEN PRINT VIEJO.FICH$;"está en formato erróneo": IF ERL=130 THEN
 RESUME 120 ELSE IF ERL=150 THEN RESUME 140 ELSE IF ERL=170 THEN RESUME 160
```

Fig. 5.

```
2020 IF ERR=56 THEN IF ERL=150 THEN RESUME NEXT ELSE PRINT VIEJO.FICH$;"no exist
e, prueba de nuevo": IF ERL=130 THEN RESUME 120 ELSE IF ERL=170 THEN RESUME 160
2030 IF ERR=65 THEN IF ERL=150 THEN PRINT VIEJO.FICH$;"ya existe. ¿Reescribo?";:
GOSUB <OBTENER SI/NO>:IF <SI> THEN RESUME NEXT ELSE RESUME 140 ELSE RESUME NEXT
```

Fig. 6.

```
2100 REM Obtener si/no
2110 CONTE$=INKEY$:IF CONTE$="" THEN 2110
2120 CONTE$=CHR$(ASC(CONTE$)AND&H5F)
2130 RETURN
la línea 2030 se sustituye por la siguiente
2030 IF ERR=65 THEN IF ERL=150 THEN PRINT VIEJO.FICH$;"ya existe. ¿Reescribo?";:
GOSUB 2100: IF CONTE$="S" THEN RESUME NEXT ELSE RESUME 140 ELSE RESUME NEXT
```

Fig. 7.

```
2040 IF (ERR=66) OR (ERR=67) THEN PRINT "Disco lleno. ¿Borro un fichero?";: GOSU
B 2100: IF CONTE$="S" THEN RESUME 2200 ELSE END
```

Fig. 8.

remos otro fichero (de consulta) para obtener información adicional que completará la ya obtenida. Finalmente, los elementos elegidos de ambos ficheros se grabarán en un tercer fichero.

En todo caso, ignoremos la rutina de la línea 1500 que es inútil, aunque sirve de ayuda para ilustrar este punto. Comprobando el posible error en el programa, no por la lógica en sí, sino obligando a leer datos erróneos, podremos obtener una pequeña tabla indicando los errores que pueden existir. Esta tabla está reflejada en la figura 1.

Cualquiera de estos errores puede llevarnos a un bloqueo del ordendor (error catastrófico), pero todos son fáciles de detectar. Hay que tener en cuenta que para obtener dicha tabla, hay que ponerse en el lugar del ordenador y esperar, siempre, el peor de los casos. Tomamos nota de todos ellos, junto con su número de código y ya

```
100 REM ****Programa Principal****
120 PRINT "Nombre del Fichero de Datos";:INPUT FICHERO.DOS$
140 PRINT "Nombre del Fichero Trabajo";:INPUT FICHERO.TRES$
160 PRINT "Nombre del Fichero de Códigos"::INPUT FICHERO.CUATRO$
180 GOSUB 1000
190 STOP
1000 REM Primera subrutina
1010 OPEN "1",2, FICHERO.DOS$
1020 OPEN "0",3, FICHERO.TRES$
1030 FOR X=1 TO 100
1040 INPUT #2,CODIGO%,INFORMACION%,CANTIDAD%
1050 GOSUB 1500
1070 NEXT X
1080 CLOSE 2
1090 KILL "VIEJO.BAK"
1100 NAME "FICHERO.DOS$" AS "VIEJO.BAK"
1500 REM
1510 OPEN "I",4,FICHERO.CUATRO$
1520 FOR Y=1 TO CODIGO%
1530 INPUT #4,NOMBRE$
1540 NEXT Y
1550 RETURN
```

SYNTAX ERROR
TYPE MISMACHT
OUT OF DATA

Listado 1

```
2000 REM **Rutina caza errores**
2010 IF (ERR=56) AND (ERL=1010) THEN PRINT "No encuentro"+FICHERO.DOS$;"Tecleelo
 de nuevo";:INPUT;FICHERO.DOS$:RESUME
2020 IF (ERR=56) AND (ERL=1530) THEN PRINT "No encuentro"+FICHERO.CUATRO$;"Tecle
elo de nuevo";:INPUT;FICHERO.CUATRO$:RESUME
2030 IF (ERR=66) OR (ERR=67) THEN PRINT "No hay sitio en el disco":PRINT "Teclee
 el fichero a borrar";:INPUT;FICHERO.A.BORRAR$:KILL FICHERO.A.BORRAR$:RESUME
2040 IF (ERR=61) AND (ERL=1040) THEN PRINT "No reconozco esos datos":PRINT FICHE
RO.DOS$;"es el correcto?";:INPUT(1);CONTES$:IF CONTES$="S" THEN PRINT "Datos err
oneos":END: ELSE <OBTENER FICHERO CORRECTO, CORREGIR ERRORES Y CONTINUAR EJECUCI
N>
2050 IF (ERR=61) AND (ERL=1530) THEN PRINT "No reconozco esos datos":PRINT FICHE
RO.CUATRO$;"es el correcto?";:INPUT(1);CONTES$:IF CONTES$="S" THEN PRINT "Datos
erroneos":END: ELSE <OBTENER FICHERO CORRECTO, CORREGIR ERRORES Y CONTINUAR EJEC
UCIN>
2060 IF ERR=65 THEN KILL "viejo.bak"
2070 IF (ERR=60) AND (ERL=1010) THEN PRINT FICHERO.DOS$;"tiene formato erroneo":
PRINT "Tecleelo de nuevo";:INPUT FICHERO.DOS$:RESUME
2080 IF (ERR=60) AND (ERL=1020) THEN PRINT FICHERO.TRES$;"tiene formato erroneo"
:PRINT "Tecleelo de nuevo";:INPUT FICHERO.TRES$:RESUME
2090 IF (ERR=60) AND (ERL=1510) THEN PRINT FICHERO.CUATRO$;"tiene formato errone
o":PRINT "Tecleelo de nuevo";:INPUT FICHERO.CUATRO$:RESUME
```

Listado 2

estamos listos para iniciar la caza y captura de errores indeseados. La tabla, siempre a mano al realizar un programa, con todos los códigos de error nos servirá para crear unas rutinas que permitirán detectar, informar y actuar debidamente, frente a los posibles problemas.

El código de error 55 se detecta mejor utilizando la función EOF en lugar de *ON ERROR*... Por ejemplo, se podrá emplear la instrucción siguiente:

IF EOF (número de fichero) *THEN* (acción)

Para el resto de los errores, lo mejor es inicializar una serie de rutinas caza-errores. Estas rutinas se pueden situar en cualquier lugar dentro del programa, sin embargo, la situación ideal es al principio de éste. En nuestro ejemplo, pondremos:

110 ON ERROR GOT 1000

Inmediatamente después de detectar un error, el programa pasa el control a la línea 2000 automáticamente, con las variables *ERR* y *ERL* debidamente preparadas. En esta línea, habrá que insertar una rutina que corrija o ignore el error producido, opción a elegir por el programador en función del programa. En algunas ocasiones (las menos), ignorar el error es muy recomendable ya que es más complicado el remedio que la enfermedad.

Por ejemplo, podemos empezar con el listado 2, pero esto lo complica demasiado aunque ejecutará correctamente, lo poco que va a ejecutar, las líneas 2030 y 2050 van a tener que tomar unas decisiones bastante complejas referentes a la acción a tomar con los datos ya ejecutados y escritos en el FICHERO.TRES$.

Seguidamente, habrá que reducir la gran cantidad de códigos duplicados entre las líneas 2060 a 2080.

A continuación, habremos llegado al fondo del problema. Estamos en el centro de una rutina caza-errores, lo que significa que NO PODREMOS DETECTAR ERRORES GENERADOS EN ESTA PARTE DEL PROGRAMA, ya que todos los errores producidos aquí son fatales.

Como estamos introduciendo datos directamente en esta rutina, podemos tener problemas y esto es lo que hay que evitar, por lo que habrá que buscar otro método.

La solución en principio es fácil. Se trata de detectar los errores en el momento de introducir datos al ordenador. Esto es la clave de todo el problema, determinar cuanto antes el error. La basura que se introduce en el ordenador, una vez haya pasado un control, podrá causar más destrozos que si el ordenador se colgara. ¿Qué podemos hacer para detectar los errores en la entrada de datos? Partiendo de nuestro ejemplo, lo primero que haremos es comprobar si el nombre del fichero es correcto y, por supuesto, si ese fichero existe, de lo contrario podemos haber borrado un fichero importante.

Empecemos nuevamente todo el proceso, partiendo de cero.

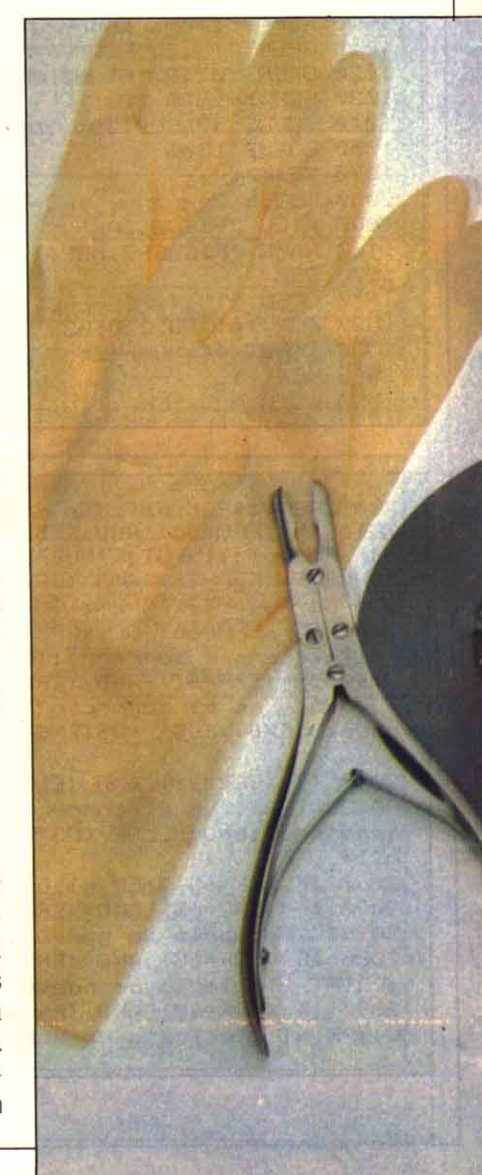
La instrucción:

NAME (nombre antiguo) *AS* (nombre nuevo)

se puede utilizar para comprobar el formato del nombre del programa. La instrucción *NAME* implica la existencia de los parámetros —nombre antiguo—, pero no la existencia del —nombre nuevo—. De no ser así, aparecerá el mensaje de error correspondiente. En nuestro ejemplo, habrá que insertar las líneas que aparecen en la figura 2.

Introduciendo una nueva variable, FICHERO.MALO$, podemos rebajar el número de códigos en nuestra rutina de control de errores. Lógicamente, para empezar, tendremos que comprobar el formato correcto del nombre del fichero (figura 3).

Ya empezamos a ser más cuidadosos, pero desafortunadamente, como no está permitida la instrucción;



RESUME ERL—10

tendremos que buscarnos la forma de engañar al ordenador (figura 4).

¿A que parece que está bien? Pues no, ésta es la clásica rutina para salir al paso de este problema, comúnmente se la denomina el síndrome de la instrucción *IF* en serie.

No es la lógica lo que está mal, es la sintaxis. Lo que ocurre es, que si el test que se propone en la primera mitad de la instrucción 2030 está mal, entonces el resto de la instrucción se ignorará completamente.



La solución está en escribir cada instrucción IF en una línea o incluir la palabra *ELSE* para asegurarse que todas las condiciones están anidadas correctamente. Es el método más sencillo de seguir y aplicar, cuyo resultado es parecido al de la figura 5.

Si después, introducimos datos erróneos, el programa automáticamente nos enviará a la rutina de entrada de datos, hasta que introduzcamos los datos correctamente.

En nuestro caso particular, dos ficheros son de entrada de datos y como tales han de existir. Sin embargo, el tercer fichero es de salida y si no existe, cuando se ejecute la línea 1020 el intérprete lo creará. Esto puede generar un problema al volver a reescribir un fichero accidentalmente, pero afortunadamente tenemos la suficiente información en ERR para evitar semejante problema y podremos comprobar la existencia de un duplicado del fichero.

Observe que debido a la complejidad se ha introducido una instrucción *GOSUB* en la rutina cazaerrores para obtener una respuesta si/no. Con frecuencia, será necesario utilizar subrutinas para detectar errores. La figura 7 muestra cómo escribir este programa.

A continuación vemos cómo es la sintaxis de la instrucción *RESUME NEXT*. En el caso de responder afirmativamente a la pregunta de reescribir el fichero, entonces la instrucción *RESUME* habrá dejado al programa dentro de un bucle infinito en la línea de error, pero hemos preferido omitir este caso y continuar con la siguiente instrucción.

Hay una cuestión importante referente al comando *GOSUB*. Las rutinas que requieren la introducción de datos por teclado se deben usar con moderación para detectar errores. La razón se debe a que una vez estemos dentro de una rutina *ON ERROR*, el resto de las rutinas de detección de errores se deshabilitan hasta que haya una instrucción *RESUME*.

Conclusión

Este ejemplo (que volvemos a insistir, es sólo un fragmento de un hipotético programa de gestión), permite afrontar los errores de otra forma y con cierta seguridad, aunque obviamente las rutinas se han de trabajar más detalladamente,

SYNTAX ERROR
TYPE MISMACHT
OUT OF DATA

El hecho de depurar los programas hacen que éstos sean más fiables.

cosa que dejaremos a los lectores. Sin embargo, dejaremos sentado unas bases a tener en cuenta, para cuando queramos realizar programas a otro nivel.

Primero, siempre usar la instrucción *RESUME* para volver *(RETURN)* al programa principal, no ponga un *RETURN* porque sí. Está claro que el ordenador obedecerá esa instrucción, pero todo el sistema elaborado de detección de errores de desinhibirá y la pila de subrutinas se verá afectada, haciendo que ésta se comporte de manera inesperada. El paso co-

errores

rrecto es poner una instrucción *RESUME* en el número de línea al final de la subrutina en la que se encuentra.

Segundo, es bueno acostumbrarse a finalizar todas las rutinas de detección de errores con el comando;

ON ERROR GOTO 0

que finaliza e imprime cualquier error no permitido. Este se convierte en una red de seguridad que no permitirá que se bloquee el ordenador en caso de existir un error que lo produzca.

Y tercero, las rutinas de detección de errores se pueden convertir en auténticos laberintos de *GOTOS,* debido a que es imposible conseguir una estructuración adecuada. Es importante mantener estas rutinas lo más simple posible y modulares, aunque haya que repetirlas en diferentes lugares del programa.

Para finalizar, mostraremos una tabla con los códigos de error y su significado y un programa que cambia los mensajes de error al castellano.

Con esta tabla y una ideas claras, se podrán crear programas de más nivel, con posibilidades similares a la de los programas serios.

NOTA: La tabla está reproducida del libro *«MSX, Guía del programador y manual de referencia»* cuyos autores son T. Sato, P. Mapstone e I. Muriel de la editorial Anaya Multimedia.

El programa, enviado por Luis Serrano Alenza desde Valencia, permite sustituir los mensajes de error por sus equivalentes en castellano. El programa está realizando en el *Spectravideo 720,* por lo que en otros ordenadores habrá que controlar la disposición de los bloques de memoria.

```
10 REM **********************************
20 REM *MENSAJES DE ERROR EN CASTELLANO*
30 REM *                                *
40 REM *L. SERRANO (C) 1985             *
50 REM **********************************
60 REM
70 REM
80 REM   SUBRUTINA QUE PERMITIRA
90 REM   MODIFICAR LOS MENSAJES
100 REM  DE ERROR
110 REM
120 CLS
130 FOR A=35000! TO 35040!
140 READ B: POKE A,B: NEXT A
150 DEF USR=35000!
160 L=USR(L)
170 DATA 243,33,0,0,17,0,144,1,0,64,237,
   176
180 DATA 62,81,211,168,33,0,144,17,0,0,1
   ,0,64,237,176
190 DATA 211,168,33,0,64,17,0,144,1,0,64
   ,237,176,201
200 REM
210 REM  MODIFICA LOS MENSAJES
220 REM
230 A=15734
240 FOR B=1 TO 36 :READ A$
250 FOR C=1 TO LEN(A$): POKE A,ASC(MID$(
   A$,C,1))
260 A=A+1
270 NEXT C
280 IF A=16337 THEN PRINT"REPASE LOS DAT
   OS ALFANUMERICOS": END
290 POKE A,0
300 A=A+1
310 NEXT B
320 DATA "NEXT SIN FOR","E.Sintáctico","
   RETURN SIN GOSUB","Sin DATA","Cita
   ilegal a función","Desbordamiento",
   "Sin memoria","Línea indefinida"
330 DATA "Fuera de rango","Vector redime
   nsionado","División por 0"
340 DATA "Modo directo ilegal","Clases d
   iscordantes","String sin espacio","
   String largo","String complejo","No
   puedo seguir","Función indefinida"
   ,"Error E/S","Comprobar error","Sin
   reanude"
350 DATA "RESUME SIN ERROR","E.Inmostrab
   le","Falta operando","Buffer debord
   ado"
360 DATA "Campo rebasado","E.Interno","
   Mal numero fichero","Fichero no enc
   ontrado","Fichero ya abierto"
370 DATA "Input pasado fin","Nombre fich
   ero incorrecto","Sentencia directa
   en fichero","Secuencial E/S solo","
   Fichero no abierto","E.Inmostrable"
380 POKE 16339,101
```

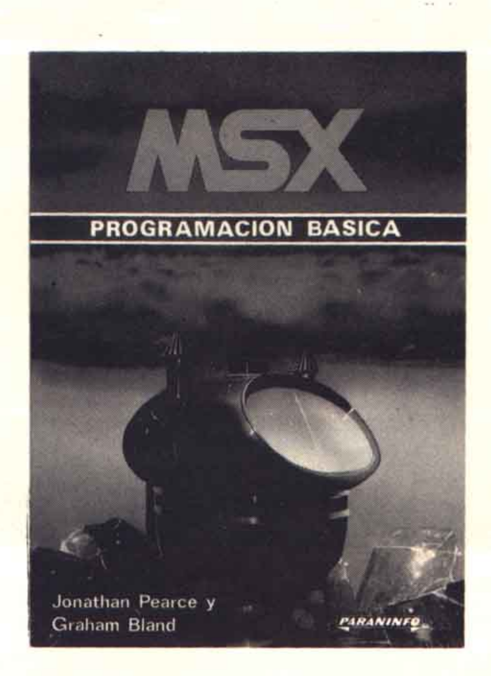


LA REVISTA IMPRESCINDIBLE
PARA LOS USUARIOS DE LOS ORDENADORES PERSONALES MSX.

Una publicación mensual que ayuda a obtener el máximo partido a su ordenador.

MSX publica cada mes programas y juegos, además de reportajes sobre programación y la posibilidad de ganar premios realizando programas y otros temas siempre de gran interés.

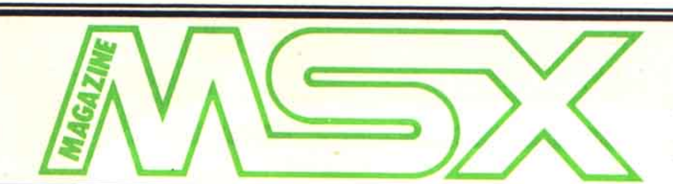




ADEMAS, **benefíciese de un 15 % DE DESCUENTO sobre el precio real de suscripción**

PRECIO NORMAL DE SUSCRIPCION	USTED SOLO PAGA	AHORRO
~~3.600~~ PTAS.	3.060 PTAS.	15 %

APROVECHE AHORA esta irrepetible oportunidad para suscribirse a **MSX**. Envie **HOY MISMO** la tarjeta adjunta a la revista, que no necesita sobre ni franqueo. Deposítela en el buzón más cercano. Inmediatamente recibirá su primer ejemplar de **MSX** más el **REGALO**. Y así durante un año (12 números).



Bravo Murillo, 377
Tel. 733 79 69
28020 MADRID

SOFTWARE

Programa: Decathlon
Tipo: Juego
Distribuidor: Proein, S.A.
Formato: Cassette



En el deporte podemos señalar una gran variedad de modalidades, pero ninguna de ellas por sí sola es completa. Esto no quiere decir que ponerse en forma no se puede conseguir practicando un solo deporte, sino que, ciertas partes del cuerpo se mueven más que otras, por lo que practicar varios deportes, o realizar diversos ejercicios, sea la mejor manera de ponerse en forma.

DECATHLON se llama a la conjunción de diversos deportes en una forma física inmejorable, pero en este deporte no podremos regatear esfuerzos. Así que ya sabe, nuestro *DECATHLON* no será de verdad, pero en el momento en que se siente frente a su ordenador tendrá que estar en su mejor momento físico y ayudarse de su mayor habilidad y concentración.

Consta de diez pruebas, de ahí el prefijo *DECA*, comenzando por las carreras de velocidad, 100 m, y siguiendo por los 400 m, hasta llegar a resistencia. Nos encontramos entre ellas con diversos deportes de lanzamiento: jabalina, disco y altura, así como el espectacular salto de vallas. Longitud y velocidad también estarán presentes en esta olimpiada doméstica, en la que encontrará gran ejercicio, sobre todo psíquico.

Las características del juego, en cuanto a realización, son sencillas, todas las pruebas se realizan en la pista de velocidad, escenario uniforme para nuestro *DECATHLON*, pero por la configuración de nuestros deportistas y los movimientos tan exactos que realizan, podemos englobarlos en el conjunto de inmejorables juegos de software en cassette.

También hemos de señalar algunas de las habilidades de este programa, como por ejemplo la falta de marcadores, no los referidos a tiempo y distancia, sino aquellos que nos marcan los récords efectuados por los anteriores competidores, ya que al no saber cómo debe ser nuestro esfuerzo, éste sea casi siempre menor.

Podremos utilizar *joystick* o cursores, pero de cualquier manera no es difícil conseguir una buena puntuación. Basta con mover los cursores de derecha e izquierda para coger velocidad y pulsar la barra espaciadora a la hora de saltar o lanzar un objeto (jabalina, disco, etc.).

Ya sabemos que todos los juegos realizados para MSX son de una gran claridad en cuanto a colores, gráficos y sonido, por lo que en este juego se ven con exactitud hasta los más mínimos detalles, los focos que alumbran el recinto del público, o los equipos de los participantes.



La distribución de los colores es inmejorable, un fondo verde que nos ayuda a distinguir de forma clara la pista de carreras en rojo y sobre todo, y más importante, los marcadores con los que contamos, el tiempo que tardamos en realizar una prueba o la distancia que hemos conseguido en los lanzamientos o en el salto.

Una de las mejores pruebas de las que consta este juego es el salto de vallas. La peculiaridad de esta modalidad es la elasticidad que realiza el *DECATHLON-man* a la hora de saltarlas y la precisión de sus saltos, claro está que todo depende de nosotros.

Y la última característica del juego, es que también pueden ser dos personas las que compitan, en este caso habrá que utilizar el *joystick*.

Ya es hora de convertirse en un *DECATHLON-man*.

Puntuación:
Presentación: 8
Rapidez: 6
Claridad: 8
Adicción: 7

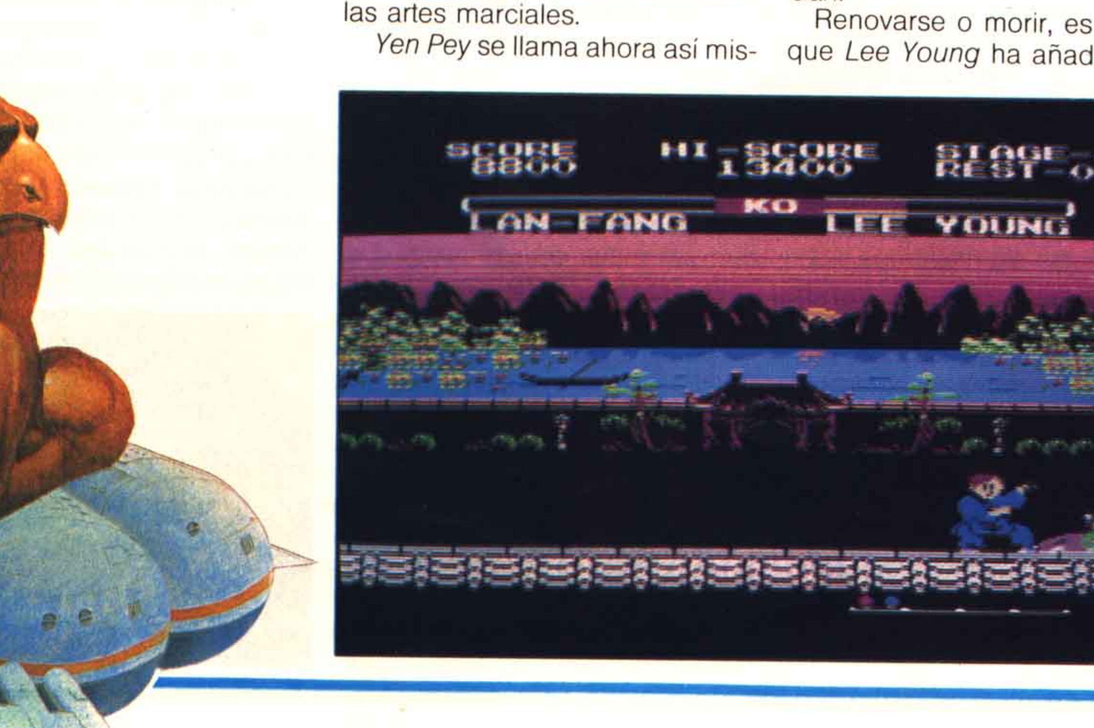
Programa: Yie-Ar Kung-Fu
Distribuidor: Serma
Tipo: Juego
Formato: Cartucho ROM

Hace veinte años que *Lee*, el maestro del *Kung-Fu*, aniquiló la banda de criminales de *Chop Suey*. Pero un miembro de la banda consiguió sobrevivir: *Yen Pey*... y ahora todos aquellos que disfrutamos con el primer *Kung-Fu*, veremos sublimados nuestros golpes a la hora de competir con los más hábiles y astutos maestros de las artes marciales.

Yen Pey se llama ahora así mismo el emperador *Yie Gah*, y conjuntamente con sus jefes militares y sus hábiles secuaces, ha extendido su diabólica influencia a lo largo de toda China. Nuestro brazo derecho, *Lee*, maestro entre maestros, ha visto frustradas sus aspiraciones debido al paso del tiempo, por eso su hijo y heredero de toda su sabiduría es quien debe enfrentarse a sus enemigos.

Siempre se dice que segundas partes nunca fueron buenas, pero aquí tenemos la excepción que confirma la regla. La claridad y rapidez de movimientos, junto con la movilidad de los personajes, indica un avance en la creación de sprites, ya no sólo en el movimiento, sino también en su configuración. Cabe destacar con gran singularidad el paisaje de fondo en el que se desarrolla la lucha, siendo diferente cada vez que pasemos de nivel, desde un atardecer en una págoda, hasta la más temible cueva, para terminar en la absorta contemplación de un templo gigante, morada del emperador *Yie Gah*.

Renovarse o morir, es por eso que *Lee Young* ha añadido nue-



SOFTWARE

vas técnicas a la hora de competir, como por ejemplo el empleo del té *(00-LONG)*, y la potencia de los noodles *CHOW-MEIN*, todas estas nuevas técnicas irán apareciendo a lo largo del desarrollo del juego, por ello tendremos que estar atentos para no perder la ocasión cuando se nos presente. Entre nivel y nivel, nos aparecerán numerosos objetos que tendremos que destruir, su número y velocidad serán siempre diferentes, por lo que debemos estar sumamente atentos porque sino nos quitarán fuerza y perderemos poder a la hora de combatir con nuestros verdaderos enemigos.

Por lo que se refiere al contenido del juego, comenzaremos por los malvados maestros que serán nuestros adversarios, el primero de ellos es débil para nuestras sutiles tácticas, pero la cadena que utiliza para golpearnos por desgracia no lo es, girará entorno a nosotros hasta que nos descuidemos y logre batirnos de un sólo golpe. El segundo —o mejor— la segunda de nuestros enemigos, es nuestra sofisticada *Gheisa*, que con sus rápidos abanicos nos hará perder fuerza cada vez que nos toque, pero casi es mejor no superarla, pues su sucesor, *Po-Chin*, lanzador de llamas que aniquilan en el instante, nos mantendrá unos minutos sin movernos si algunas de estas llamas nos roza, después,... una innovación de *Konami*, dos adversarios a la vez, un enorme luchador y su máscara, que esperan el momento más oportuno para batirnos y dejarnos sin sentido. Lo importante en este nivel, es conseguir golpear a nuestro adversario antes de que éste consiga quitarse la máscara, pero para conseguir esto no podremos descuidarnos ni un sólo momento.

Si logramos pasar todos estos niveles nos encontraremos en el gran templo de *Yie-Gah* y la lucha será casi interminable, pero no os vamos a contar. más, con lo que les ha costado llegar a nuestro equipo, vale la pena que vosotros os esforcéis también, sino no es divertido.

Acción, seguridad y concentración son importantes a la hora de jugar. Os recomendamos que antes de poneros a jugar practiquéis los diversos golpes con vuestro *joystick*, o cursores, porque un sólo golpe puede ser decisivo.

Puntuación:
Presentación: 9
Claridad: 7
Rapidez: 7
Adicción: 9

Programa: QH, con pronóstico
Tipo: Juego
Distribuidor: SERMA
Formato: Tarjeta ROM

El mundo de las carreras de caballos es un singular deporte en el que se encuentran reunidos los más diversos personajes y los más variados ambientes.

Pero en el interior de este gran mundo no sólo el deporte y la competitividad imponen las reglas, sino que junto a ellos aparece el juego y las apuestas. Poder hacerse millonario en un instante es cuestión de saber hacer un buen pronóstico de la carrera y tener un poco de buena suerte.

Ahora bien, tenemos que diferenciar las apuestas realizadas en el hipódromo de las apuestas que se realizan con el que comúnmente llamamos boleto de la *QH*, al que va referido este programa.

El programa consta de dos partes simultáneas, una referida a estadísticas, en la que no podremos contar con la suerte, ya que los baremos que se utilizan son hechos basados en las anteriores carreras y sus resultados.

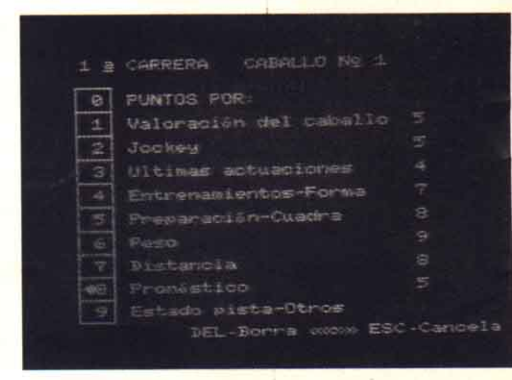
Dependiendo de la temporada de carreras y del número de las mismas hay dos tipos de estadísticas para cada caballo.

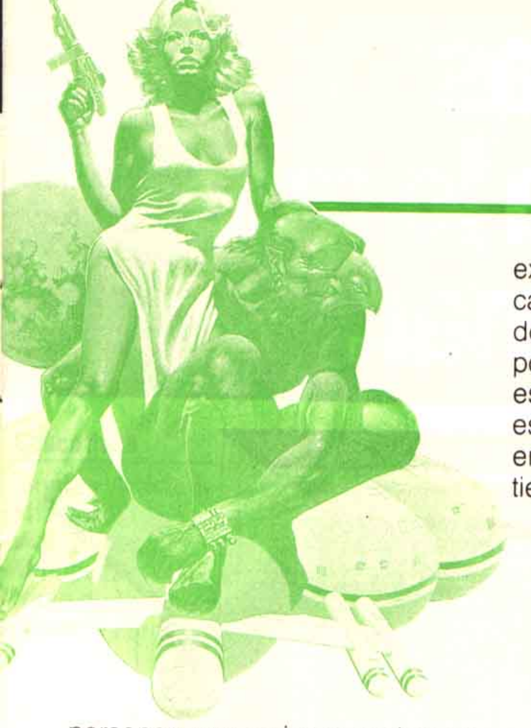
Estadística 1.ª, porcentaje (%) de veces que el caballo correspondiente ha ganado en la carrera que se está procesando.

Estadística 2.ª, porcentaje (%) de veces que el caballo correspondiente ha ganado en las seis carreras.

En este primer apartado del juego podemos hacer notar el número de elementos que necesitamos para hacer una predicción exacta de los acontecimientos, aunque en el fondo todo depende del azar.

Un control exhaustivo de las carreras realizadas a lo largo del período, es absolutamente necesario para estudiar a los caballos y las posibilidades de cada uno de ellos, así como el resultado total de las carreras. Elaborar nuestra pequeña estadística es sin duda necesario, por lo que este programa no es adecuado para aquellas





personas que quieren probar su suerte por primera vez.

La segunda parte a la que hemos hecho referencia anteriormente, es la singularización de las ventajas e inconvenientes de cada caballo en sí.

Una vez seleccionada la carrera y el número de caballos de la misma, se puntúa a cada caballo por nueve conceptos diferentes que aparecen en pantalla secuencialmente, como pueden ser la velocidad, el sprint, temperamento y aptitudes del caballo para ese tipo de carrera, ya que puede ser de velocidad, resistencia o el sueño de todo *jockey,* correr el *Gran National.*

El número de puntos obtenido por cada caballo, se combina con las estadísticas anteriormente elaboradas, proporcionando así el pronóstico, con el que sin duda acertará. Si se puntúa caro a todos los caballos se conseguirá el «pronóstico estadístico». El resultado, puntuación, estadística y pronóstico para cada caballo se produce por pantalla, o si lo prefiere por impresora. De todas formas, es la mejor forma de acertar sobre seguro.

No se trata de que este programa nos vaya a hacer millonarios en una temporada, ya que también necesita de datos científicos exactos, como son las características de los caballos y los resultados obtenidos a lo largo de la temporada, pero sin dejar de lado la estadística, un 75% de la suerte está con Ud., por eso ya es hora de empezar sus cálculos, no pierda tiempo, aún puede ser millonario.

Puntuación:
Presentación: 7
Rapidez: 7
Claridad: 8
Adicción:

Programa: Computer Hits
Tipo: Juego
Distribuidor: ABC Soft
Formato: Cassette

La extensa gama de juegos de software que hay en el mercado, hace muy difícil escoger cuál es de ellos el más interesante o el más adecuado a nuestras necesidades. *Computer Hits,* es un compendio de programas, dedicados a diversión y ocio, en el que se ha intentado dar gusto a todas las aficiones de los adictómanos al ordenador.

Comenzando por un juego dedicado al deporte, seguido de un entretenido billar en todas sus modalidades hasta terminar por el ya habitual y siempre presente juego de marcianitos, *Computer Hits* es la unión de grandes juegos en un reduciro espacio, ya que aunque éste es un tema de segundo orden también hay que mirar el espacio, porque actualmente donde cabe uno no caben seis.

Seis juegos de distintas características son los que están presentes en esta cassette. Vamos a comentar cada unos de ellos y mencionaremos las ventajas del conjunto en sí:

Brian Jacks Superstar, en él se encuentran las modalidades más sobresalientes del deporte olímpico, tanto es así que podremos disfrutar de carreras de velocidad, tiro con arco, carrera de bicicletas (aunque éste no sea un deporte olímpico), natación y como no el fútbol. Todos ellos cuantan a la hora de clasificarse y un error puede alejarnos de la competición. El sistema de juego es muy sencillo, pero de todas formas recomendamos la utilización de *joystick* para adquirir mayor velocidad en los movimientos. Se trata de ir superando cada una de las pruebas para poder pasar a la siguiente, pero con la advertencia de que ya hay unas marcas preestablecidas, que debemos superar. Los colores no son muy atractivos y los gráficos son bastantes elementales, pero esto no quita que la unión de estas alternativas hagan de este juego un ejemplar interesante a la hora de divertirse.

Disc Warrior, es el androide por excelencia, es un juego singular, no por su concepción, pues es bastante fácil de realizar, sino por el tema que trata.

Seremos un androide que vagará perdido por un gran laberinto de tres dimensiones, en el que nos atacarán seres de diversas clases, pero ninguno de forma humana por lo que aparecerán y desaparecerán con gran facilidad, por ello deberemos estar muy atentos, pues si logran batirnos acabará el juego.

Podremos movernos en todas las direcciones y disparar, pero nuestro principal fin es conseguir salir del laberinto, lo que sigue



PROGRAMAS SONY MSX

Educativos

- Monkey Academy
- Alfamat
- Viaje Espacial
- Multipuzzle
- Noria de Números
- Corro de Formas
- Coconuts
- Yo Calculo
- Selva de Letras
- El Cubo Informático
- Electro-graf
- El Rancho
- Teclas Divertidas
- Boing Boning
- Compulandia
- Mil Caras
- Logo
- Países Mundo-1
- Países Mundo-2
- Tutor
- Computador
- Adivino
- Aprend. Inglés-1
- Aprend. Inglés-2
- Cosmos
- Curso de Básic
- Juego de Números

Juegos

- Antártic Adventure
- Athletic Land
- Sparkie
- Juno First
- Car Jamboree
- Battle Cross
- Crazy Train
- Mouser
- Computer Billiards
- Alí Babá
- Track & Field-I
- Track & Field-II
- Dorodon
- Chess (Ajedrez)
- Senjo
- E.I.
- Lode Runner
- Super Tennis
- Backgammon
- Super Golf
- Hustler
- Binary Land
- Driller Tanks
- Stop the Express
- Ninja
- Les Flics
- La Pulga
- The Snowman
- Cubit
- Pack 16K
- Fútbol
- Kung Fu
- Batalla Tanques
- Mr. Wong
- Xixolog
- Buggy
- Sweet Acorn
- Peetan
- Jump Coaster
- Buggy 84
- 3D Water Driver
- Pinky Chase
- Wedding Bells
- Fightting Rider

Aplicación

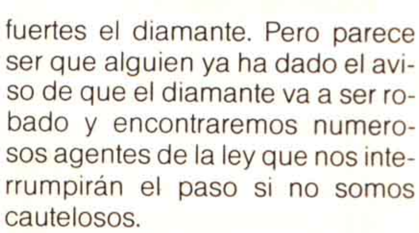
- Memoria Ram 4 K
- Creative Greetings
- Character Collect
- Quinielas y Reducciones
- Pascal
- Ensamblador
- Generador Juegos

Gestión

- Hoja de Cálculo
- Homewriter
- Control Stocks
- Contabilidad Personal
- Ficheros
- Procesador de Textos
- Control Stocks
- Vencimientos
- Contabilidad 1.500

después, esperamos que lo descubras por tí mismo.

Hustler es uno de los mejores programas que se encuentran en esta cassette, no por el tema que trata, el billar, sino por su realización, gráficos y sencillez de funcionamiento. Si sabéis las reglas del billar y sus modalidades, poco podremos contaros de éste juego, pero debemos mencionar que es muy completo y muy espectacular en cuanto a sus colores. La ejecución del juego no tiene mayor dificultad, pero ser hábil a la hora de golpear la bola será fundamental para no errar el tiro. Se puede jugar la partida contra el ordenador o bien contra un compañero. Es un buen juego.



Chukie Egg, un singular juego que no tiene como tema una guerra, aventura o una adivinanza, es un sencillo juego de habilidad, que se desarrolla en una granja que nosotros deberemos cuidar del acecho de otros animales que quieran destruirla, pero también hay que tener cuidado con los habitantes del interior de la granja, pues los temibles pollitos que viven allí, intentarán comerse el grano que hemos dejado por los rincones sin darnos cuenta de que ha ido desapareciendo.

De todas formas, este juego ya os sonará familiar porque ha sido comentado anteriormente en otro número.

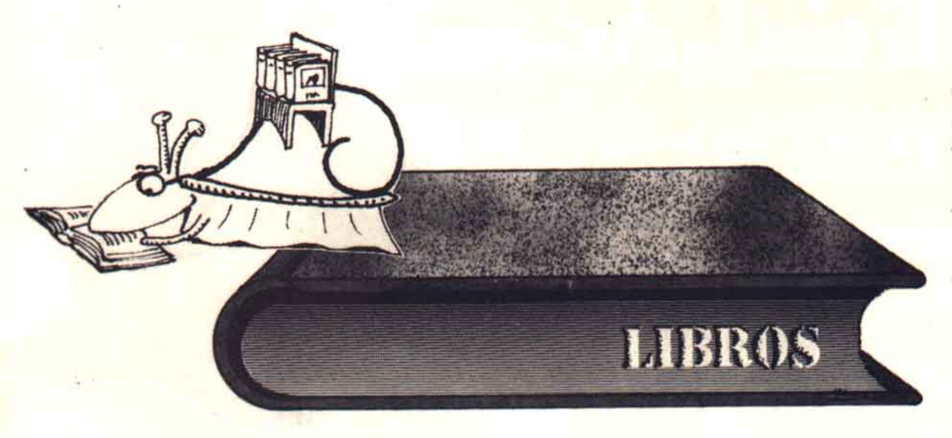
Les Flicks, podríamos mencionarlo como un juego verdaderamente innovador y espectacular por el tema que trata, la claridez de las instrucciones y el objetivo o misión del juego, que consiste en encontrar un diamante, pero encontrarlo de una manera muy poco ética, ya que nosotros seremos un ladrón.

Iremos por toda la ciudad asaltando las casas que creamos pueden tener escondidas en sus cajas fuertes el diamante. Pero parece ser que alguien ya ha dado el aviso de que el diamante va a ser robado y encontraremos numerosos agentes de la ley que nos interrumpirán el paso si no somos cautelosos.

Special Oparations, es una batalla galáctica donde los gráficos de alta resolución que nos muestra, el efecto de la perspectiva y de la velocidad, han sido creados de una manera especial por la conjunción de rayas que vienen hacia nosotros, haciendo más real esta sensación.

En cuanto a la nave de que disponemos, es un modelo bastante avanzado dentro de las características de las naves usuales, tendremos que disparar con ella a todo objeto que nos interrumpa el paso o nos dispare, así como lograr superar las sucesivas pantallas a lo largo de las que nos iremos moviendo.

Acumular puntos y entretenernos es el verdadero objetivo de nuestra misión y *Computer Hits,* ha hecho un esfuerzo por conseguirlo.



Título: Mini Diccionario Microinformático
Autor: R. Tapias
Editorial: Noray, S.A.
Páginas: 192

Generalmente, cuando alguien se introduce en el mundo de la informática, empieza hojeando revistas o libros más o menos especializadas. Pero, ya desde el principio, choca con un obstáculo difícil de superar, consistente en el léxico utilizado. A menudo se encuentran palabras y expresiones corrientes pero utilizadas en un contexto totalmente diferente y con un significado distinto al habitual. En otros casos, aparecerán totalmente nuevas, adaptaciones de palabras inglesas, que no suelen figurar en los diccionarios de lengua castellana.

En esta obra, especialmente pensada para quien se inicia en estos temas, se ha tratado de incluir estos conceptos y expresiones, con una explicación clara y detallada de su significado, de manera que la lectura de cualquier texto de informática se convierta en algo asequible para cualquier persona, por ligeros que sean sus conocimientos previos sobre el tema.

También se incluyen algunos términos coloquiales pertenecientes al argot informático que se utilizan entre los profesionales, pero que raramente se encuentran impresos. A pesar de ello, este libro no pretende decírselo todo con detalle. No se trata de un libro de texto, por lo que no incluye lecciones magistrales. Naturalmente, tampoco está dedicado al profesional especializado en informática, entre otros motivos porque éste ya utiliza, y por lo tanto conoce, las expresiones y palabras explicadas en el libro. Es simplemente una guía a la que puede recurrir el aficionado para orientarse en un terreno que posiblemente todavía no le resulte demasiado familiar.



El libro consta de dos partes: el diccionario, propiamente hablando, con los términos en castellano y un anexo en el que se incluye un pequeño vocabulario inglés/castellano con las palabras más corrientes. Al contrario de otros diccionarios, que utilizan como referencia las palabras originales en inglés, se ha preferido relacionarlas en castellano pensando en el usuario medio, que habitualmente intentará introducirse en la informática a través de textos en castellano, para pasar más adelante, una vez dominado el tema, a hojear algún texto en inglés. En definitiva, una obra interesante para aquellos que empiezan.

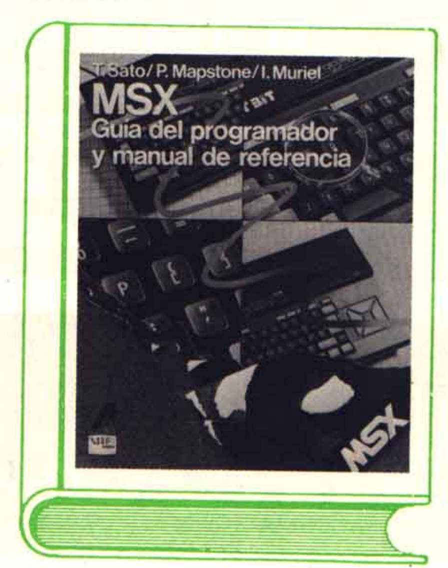
Título: MSX - Guía del programador y manual de referencia
Editorial: Anaya Multimedia
Autores: T. Sato, P. Mapstone e I. Muriel
Páginas: 702

De todas las gamas y tipos de libros vistos, leídos y comentados, podemos afirmar sin temor a equivocarnos, que estamos ante la guía más completa del sistema MSX del mercado. En ella podemos encontrar la información técnica necesaria para aprovechar al máximo las características del MSX, tanto internas como externas, es decir, desde las instrucciones (se comentan absolutamente todas), hasta los *port* de *joysticks* pasando por el interface *PPI*. A nuestro modesto entender es una compra obligada, ya que reune en un sólo volumen todo tipo de explicaciones acerca del uso de instrucciones concretas, *port* de cassettes, paginación de memoria, etc.

Este libro se encuentra dividido en cuatro partes, a saber;

Introducción al *BASIC* del MSX,

guía avanzada de programación, sección de consulta y sistema operativo. Evidentemente, la primera de ellas es una completa introducción al estándard, sus instrucciones, teclas de control, etc., donde encontraremos explicado, con todo lujo de detalles, el juego completo de instrucciones y sus aplicaciones, así como el editor de pantalla.



A continuación viene la guía avanzada de programación, donde hallaremos todo lo relacionado con este tema. Este capítulo es importante, ya que se explica con profundidad el procesador de vídeo y el *chip* generador de sonidos. Es importante destacar la profusión de ejemplos que se han utilizado en esta sección del libro.

No se deja nada de lado, ni se da nada por sentado, algo que hemos podido constatar en algunas publicaciones. Prueba de ello es que, además de tratar las instrucciones, también se analizan los *port* del ordenador, desde el conector para el cassette hasta el de *joystick*, pasando por el conector de cartuchos *ROM*.

Las dos partes siguientes son dignas de tener en cuenta. La primera de ellas son las instrucciones del *BASIC* en orden alfabético y la segunda es el sistema operativo.

En la primera parte se explican las instrucciones del *BASIC* en orden alfabético, lo cual puede ser de ayuda en aquellos momentos en que tengamos que echar una mano del manual para saber de que trataba tal o cual instrucción. Aquí, además de explicar dicho comando, se nos indicará en que capítulo podremos encontrarlo y en qué página.

Para finalizar, nada mejor que hacerlo con lo que es el sistema operativo. Ningún libro analizado hasta la fecha traía tan importante sección, del cual sería digno mencionar todo en su conjunto, pero por razones obvias de espacio no podemos hacerlo.

En suma, se trata de un libro indispensable en cualquier biblioteca y para cualquier adicto al MSX.

Título: MSX, Gráficos y Sonido
Autor: Lüers
Editorial: Ferre Moret
Páginas: 463

De todos es sabido las increíbles posibilidades gráficas y sonoras de los ordenadores MSX, pero pocos son los que en realidad manejan y sacan el auténtico provecho de tales cualidades. Desgraciadamente, el usuario medio encuentra que en los manuales que acompañan al ordenador hay poco o nada explicado sobre este sistema y se ve en la necesidad de tener que echar mano de manuales adicionales o libros, que frecuentemente encontrará en inglés.

Este libro analiza detenidamente todas las posibilidades, tanto gráficas como sonoras del estándar MSX, con ejemplo y 16 programas completos.

A su vez, se van analizando todas las instrucciones gráficas y sonoras (que no son pocas) de este *BASIC* y particularmente los más complejos, como *SOUND*, *BASE* y *VDP*. La explicación de los comandos va acompañada por una serie de programas ejemplo que facilitan la comprensión de estos y permiten aplicarlos a unos casos concretos. A continuación, se dan una serie de consejos y trucos que permiten aprovechar



esas cualidades que el ordendor posee, así como una descripción de algunas rutinas en código máquina y las variables del sistema relacionadas con los gráficos y el sonido.

Al final del libro, viene una tabla con las direcciones más importantes de la *RAM* y *ROM*.

El libro está orientado tanto a los usuarios que posean los conocimientos básicos, como para aquellos programadores con experiencia en el sistema MSX. Está escrito en un lenguaje sencillo y claro, sin demasiados tecnicismos, lo que facilita enormemente su comprensión. Sin embargo, como punto negativo está la terminación del producto, siendo ésta algo pobre y no muy trabajada.

Spectravídeo

Este ordenador representa una clara superación del modelo anteriord, al que indudablemente sustituirá. Todo lo que este ordenador incorpora lo convierte en el aparato más competitivo del mercado. Entre las distintas ventajas que dispone, hemos de destacar la incorporación de una unidad de discos de 3.5 pulgadas, así como dos diskettes sumamente importantes que dan al ordenador un aspecto más semi-profesional que el resto de la gama e incluso que ordenadores de su misma generación, ya que algunos aparatos de la 2.ª Generación vienen con este tipo de *software*. Se trata del *MSX-DOS* y del *CP/M*, que en este ordenador corren de maravilla. Sin embargo, el sistema operativo *CP/M* no va con otros ordenadores MSX, lo que pone en peligro la compatibilidad del estándar.

Pero pasemos a los aspectos más relevantes de este nuevo aparato.

El ordenador es muy compacto y pesado, lo que da una sensación de solidez que no existe en otras máquinas. El teclado es bueno, pero demasiado abrumador. Todas las teclas están juntas y es preciso acostumbrarse al tacto y a la sensación que da el estar todo al alcance de la mano, apenas hay que desplazar los dedos para llegar a las teclas más comprometidas. Todo queda cerca, desde las teclas del cursor, hasta las de función.

La unidad de discos se encuentra situada en la parte lateral derecha del ordenador, con un *led* encima de las teclas del cursor que indica el estado de funcionamiento. El ordenador, una vez soltado el asa que cubre la parte posterior que protege todos los conectores, tiene un ángulo de inclinación que resulta muy cómodo a la hora de escribir. Dichos conectores están bien protegidos y sin embargo, una vez conectados todos los cables, permanecerán muy accesibles.

El completo juego de conectores, incluyen desde el *interface* Centronics hasta el *RS-232C*, siendo el único ordenador (junto con el Toshiba *HX-22*) que lo incorporan en sus modelos (en algunos ordenadores de la 2.ª Generación incorporan el euroconector).

El resto de las conexiones, cassette y *joysticks*, se encuentran en la parte lateral derecha, a continuación de la unidad de disco.

Por contra, este ordenador no trae *software* incorporado en *ROM*, pero sí se vende con los dos discos mencionados anteriormente y con una serie de manuales que completan todo el paquete, cuyo precio es más que razonable.

La posibilidad de utilizar las 80 columnas, hacen que el *CP/M* corra sin ninguna dificultad. Aún nos queda por ver qué programas estarán asequibles para el estándar en este sistema operativo, ya que por el momento no hay nada seguro.

Volviendo a las características del ordenador, tenemos ante nosotros el primero que se comercializa con una elegante bolsa de viaje, lo que facilita enormemente su traslado y manejo. En la bolsa





cabe de todo, desde los numerosos manuales que trae el ordenador hasta cartuchos *ROM* o diskettes si llega el caso. Aún con todo, es importante tener en cuenta que no tiene la fuente de alimentación incorporada como era de esperar, pero es normal, ya que en su lugar tenemos una unidad de discos, algo que en el fondo agradecerá el potencial usuario de este aparato que sin lugar a dudas se convierte en la mejor baza de los ordenadores MSX. Su precio está por debajo de las 85.000 ptas. y si tenemos en cuenta todo lo que se nos ofrece y hacemos números, no cabe la menor duda de que, hoy por hoy, es una de las mejores opciones por el momento.

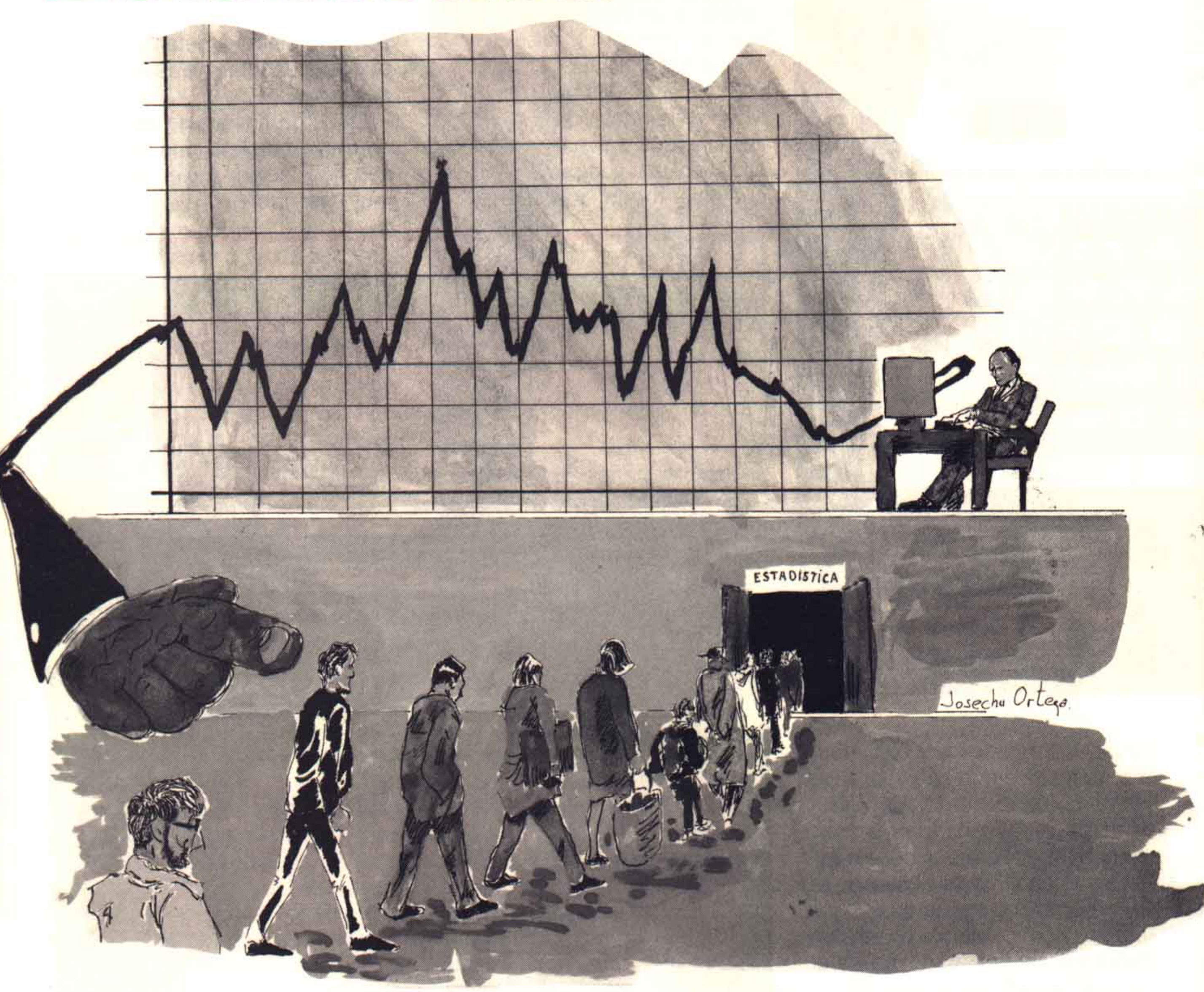
Complementando el equipo, vienen cinco manuales muy completos y detallados, cada uno dedicado a un tema concreto. Estos son, el Manual del Usuario, Manual del MSX DOS, Guía de usuario de *CP/M*, Manual del *BASIC MSX* y Manual del *Disk BASIC*. Lástima que salvo el primero de ellos, el resto venga en inglés. De cualquier modo, aquel que domine este idioma, comprobará que los manuales están bien escritos, son completos y detallados, aunque su presentación deje algo que desear.

En suma, se trata de una máquina muy completa, con unas buenas prestaciones y a un precio muy razonable.

Foto 1. Vista posterior de los conectores que incorpora el SVI-X'PRESS, donde se puede apreciar el RS-232 C.

Foto 2. La unidad de disco de 3.5 pulgadas es uno de los factores más determinantes de esta nueva versión.

Estadística



El siguiente programa, aplicado al análisis estadístico unidimensional (sobre una sóla variable), es un ejemplo ilustrativo de la gran ayuda que nos puede prestar nuestro ordenador en el campo estadístico.

A la hora de estudiar el comportamiento de una variable, deberemos tener presente la cantidad de observaciones que sobre la misma realizamos, así como el número de valores distintos que toma la variable. Es por ello que, para comenzar el análisis, el programa parte de un MENU que ofrecerá al usuario el tipo de distribución más adecuado a la muestra recogida.

En el caso de ser pocas las observaciones realizadas no se hará necesario el agrupamiento previo de los datos y así nos encontraríamos ante una distribución del «tipo I». Obteniéndose como resultados aquellas medidas de concentración y dispersión más importantes.

Para el caso de ser muchas las observaciones realizadas sobre la variable, deberemos considerar si

SERVICIO DE EJEMPLARES ATRASADOS

ESTOS SON LOS EJEMPLARES DE MSX MAGAZINE APARECIDOS EN EL MERCADO CON UN RESUMEN DE SU CONTENIDO



Núm. 1
¿Qué es el MSX? Su BASIC, periféricos, programas, software.



Núm. 2
Generación de sonido. MSX-DOS, el ordenador por dentro, programas, noticias.



Núm. 3
Los joysticks, 256 caracteres programables, Z80 corazón de león, compro/vendo/cambio.



Núm. 4
Las comunicaciones entre ordenadores, la jerga informática, trucos, rincón del lector.



Núm. 5
Comandos de entrada/salida, el BASIC MSX comparado con Spectrum y Commodore 64. Código Máquina.



Núm. 6
Los 8 magníficos (test gigante), el bus de expansión, los misterios de la grabación, programas.



Núm. 7
Analizamos el Generador de Sonido. Aplicaciones matemáticas con el ordenador. La memoria de video. Trucos, noticias.



Núm. 8
Compact Disc, el periférico del futuro. Test: Dynadata DPC-200. Continuamos con la memoria de video. Libros, software, programas, trucos.



Núm. 9
Características técnicas del Compact Disc. Tratamiento de datos. Test: Quick Disk. Trucos, libros, noticias, programas.

PARA HACER SU PEDIDO, RELLENE ESTE CUPON, HOY MISMO Y ENVIELO A MSX MAGAZINE BRAVO MURILLO, 377. Tel. 733 96 62 - 28020 MADRID

Ruego me envien los siguientes números atrasados ....

al precio de **250** ptas. cada uno. Cuyo importe abonaré:

☐ POR CHEQUE ☐ CONTRA REEMBOLSO ☐ CON MI TARJETA DE CREDITO

☐ AMERICAN EXPRESS ☐ VISA ☐ INTERBANK

Número de mi tarjeta ....

Fecha de caducidad ....

NOMBRE ....

DIRECCION ....

POBLACION .... C.P. ....

PROVINCIA ....

los valores que ésta toma se repiten con mucha o poca frecuencia. Si el número de valores que toma la variable no sobrepasa 20 nos encontraremos en el caso ideal para optar por una distribución del «tipo II». Por el contrario, nos situaremos en una distribución del «tipo III», cuando el campo de variación de la variable sea demasiado grande. Procederemos en este último caso a la agrupación en intervalos de los valores observados.

Para mayor claridad supongamos que realizada una encuesta sobre la edad de los lectores de esta revista sólo han respondido a la misma 50 personas (la muestra desde luego no sería muy significativa), estaríamos entonces ante una distribución del «tipo I», iríamos introduciendo las edades de los encuestados de una a una.

Suponiendo que el número de los encuestados fuese 2700 y sus edades estubiesen comprendidas entre 15 y 30 años ($30 - 15 \leq 20$), efectuaríamos previamente un agrupamiento de los valores observados como muestra la siguiente tabla:

EDAD	PERSONAS
15	12
16	25
17	53
18	100
19	150
20	275
21	295
22	311
23	309
24	272
25	195
26	173
27	155
28	150
29	115
30	110

Si en el caso anterior hubiésemos encontrado la edad de los lectores comprendida entre 10 y 50 años procederíamos a agrupar estas edades en intervalos. Naturalmente, el número y amplitud de intervalos es optativo respetando un máximo de 20 intervalos.

EDAD	PERSONAS
10 - 15	155
15 - 20	245
20 - 25	450
25 - 30	580
30 - 35	510
35 - 40	355
40 - 45	255
45 - 50	1150

Por razones de presentación en la pantalla de los resultados, los valores de las variables, en los tipos II y III, deberán estar comprendidos entre –9999.99 y 9999.99. No siendo esto obstáculo para estudio de variables con otras dimensiones. Así, por ejemplo, si la variable a estudiar fuese pesetas y los valores observados superiores, bastaría considerar la variable en unidades de miles de pesetas y proceder al análisis.

La terminología utilizada en la presentación de los resultados es la más usual en los libros de estadística siendo:

li	El límite inferior del intervalo.
Li	El límite superior del intervalo.
Xi	Valor de la variable o marca de clase.
ni	Frecuencia absoluta simple.
Ni	Frecuencia absoluta acumulada.
fi	Frecuencia relativa simple.
Fi	Frecuencia relativa acumulada.
di	Densidad del intervalo (tipo III).
n	Número de valores observados.
k	Número de valores diferentes (tipo II) o intervalos (III).

Eduardo M. Díaz
Avilés (Asturias)

```
10 '**ANALISIS ESTADISTICO**
20 '  E. Diaz de la Campa
30 'M.S.X.       Enero-85
40 '
50 '
60 ' **Inicializaciones**
70 KEY OFF:COLOR 15,1,1:CLS
80 DIM X(19),N(19),NA(19),F(19)
   ,FA(19),LI(19),LS(19),D(19
   )
90 ON ERROR GOTO 3990
100 T$="Pulse tecla para contin
    uar"
110 D$=STRING$(32,"≡")
120 '
130 '
140 ' **Menu Principal**
150 SCREEN 2
160 LINE(40,0)-(216,24),4,BF
170 LINE(40,40)-(216,176),4,BF
180 OPEN "grp:"AS#1
190 PSET(48,8),4:PRINT#1,"ANALI
    SIS ESTADISTICO"
200 RESTORE 310
210 FOR I=48 TO 160 STEP 16
220 READ A$
```

```
230 PSET(56,I),4:PRINT#1,A$
240 NEXT I
250 CLOSE#1
260 A$=INPUT$(1)
270 IF A$<"1" OR A$>"4" THEN BE
    EP:GOTO 260
280 C=0:S=0:H=0:P=1:W0=0:W1=0:W
    2=0
290 ON VAL(A$) GOSUB 350,690,98
    0,320
300 GOTO 150
310 DATA "    DISTRIBUCIONES",1.
    -DEL TIPO I,2.-DEL TIPO II
    ,3.-DEL TIPO III,4.-FIN DE
    L ANALISIS,,,"        Pulse
    Opción ?"
320 SCREEN0:WIDTH32:LOCATE10,10
    :PRINT"HASTA LUEGO":END
330 '
340 '
350 '       **Tipo I**
360 SCREEN 0:WIDTH 32
370 LOCATE 1,5:PRINT "Introduzc
    a los datos a estudiar"
380 LOCATE 5,24:PRINT "Entre F
    para terminar"
390 LOCATE 10,10:INPUT"VALOR :"
    ;A$
400 X=VAL(A$):X1=X:X2=X
410 C=C+1:S=S+X
420 P=P*X
430 H=H+1/X
440 IF X>X1 THEN X1=X
450 IF X<X2 THEN X2=X
460 LOCATE 19,10:PRINT SPACE$(L
    EN(A$))
470 LOCATE 17,10:INPUT A$
480 IF A$="f" OR A$="F" THEN 50
    0
490 X=VAL(A$):GOTO 410
500 CLS
510 PRINT TAB(10);"RESULTADOS"
520 PRINT D$:PRINT"MEDIAS":PRIN
    T
530 PRINT USING"  Aritmética...
    .:#.######^^^^";S/C
540 IF W1=1 OR (P<0 AND CMOD2=0
    ) THEN 560
550 PRINT USING"  Geométrica...
    .:#.######^^^^";P^(1/C)
560 IF W2=1 THEN 580
570 PRINT USING"  Armonica.....
    .:#.######^^^^";C/H
580 PRINT D$:PRINT"DISPERSIONES
    ":PRINT
590 PRINT USING"  Valor Máximo.
    .:#.######^^^^";X1
600 PRINT USING"  Valor Mínimo.
    .:#.######^^^^";X2
610 PRINT USING"  Recorrido....
    .:#.######^^^^";X1-X2
620 IF X2=0 THEN 640
630 PRINT USING"  Coef. apertur
    a:#.######^^^^";X1/X2
640 PRINT USING"  R. Relativo..
    .:#.######^^^^";(X1-X2)/(S
    /C)
650 PRINT D$:PRINTTAB(3);T$;
660 A$=INPUT$(1):RETURN
670 '
680 '
690 '       **tipo II**
700 A$="variables":GOSUB 3400
710 FOR I=0 TO K
720 PRINT "Variable n:";I+1
730 INPUT " Valor.....:";X(I)
740 INPUT " Frecuencia:";N(I)
750 C=C+N(I):PRINT
760 NEXT I
770 ' ordenacion
780 FOR I=0 TO K-1
790 FOR J=I TO K
800 IF X(I)<X(J)THEN 820
810 SWAP X(I),X(J):SWAP N(I),N(
    J)
820 NEXT J,I
830 GOSUB 3500 'cálculos
840 ' mediana
850 H=C/2
860 FOR I=0 TO K
870 IF NA(I)=H THEN ME=(X(I)+X(
    I+1))/2:GOTO 900
880 IF NA(I)>H THEN ME=X(I):GOT
    O 900
890 NEXT I
900 ' moda
910 H=N(0):MO=X(0)
920 FOR I=1 TO K
930 IF N(I)>H THEN H=N(I):MO=X(
    I)
940 NEXT I
950 B$="II":GOTO 1410 'menuII
```

```
960 '
970 '
980 '        **tipo III**
990 A$="intervalos":GOSUB 3400
1000 FOR I=0 TO K
1010 PRINT "Intervalo n:"I+1
1020 IF I>0 THEN LI(I)=LS(I-1):
     GOTO 1040
1030 INPUT "Limite inferior";LI
     (I)
1040 INPUT "Limite superior";LS
     (I)
1050 INPUT "frecuencia.....";N(
     I)
1060 C=C+N(I):X(I)=LI(I)+(LS(I)
     -LI(I))/2:PRINT
1070 NEXT I
1080 ' ordenacion
1090 FOR I=0 TO K-1
1100 FOR J=I TO K
1110 IF X(I)<X(J)THEN1140
1120 SWAP X(I),X(J):SWAP N(I),N
     (J)
1130 SWAPLI(I),LI(J):SWAPLS(I),
     LS(J)
1140 NEXT J,I
1150 GOSUB 3500'cálculos
1160 '  mediana
1170 H=C/2
1180 FOR I=0 TO K
1190 IF NA(I)=H THEN ME=LS(I):G
     OTO 1220
1200 IF NA(I)>H THEN ME=LI(I)+(
     H-NA(I-1))/N(I)*(LS(I)-LI(
     I)):GOTO 1220
1210 NEXT I
1220 '  moda
1230 H=LS(0)-LI(0)
1240 FOR I=1 TO K
1250 IF LS(I)-LI(I)<>H THEN W0=
     1:GOTO 1270
1260 NEXT I
1270 FOR I=0 TO K
1280 IF W0=0 THEN D(I)=N(I):GOT
     O1300
1290 D(I)=N(I)/(LS(I)-LI(I))
```

LOS JUEGOS ELECTRONICOS

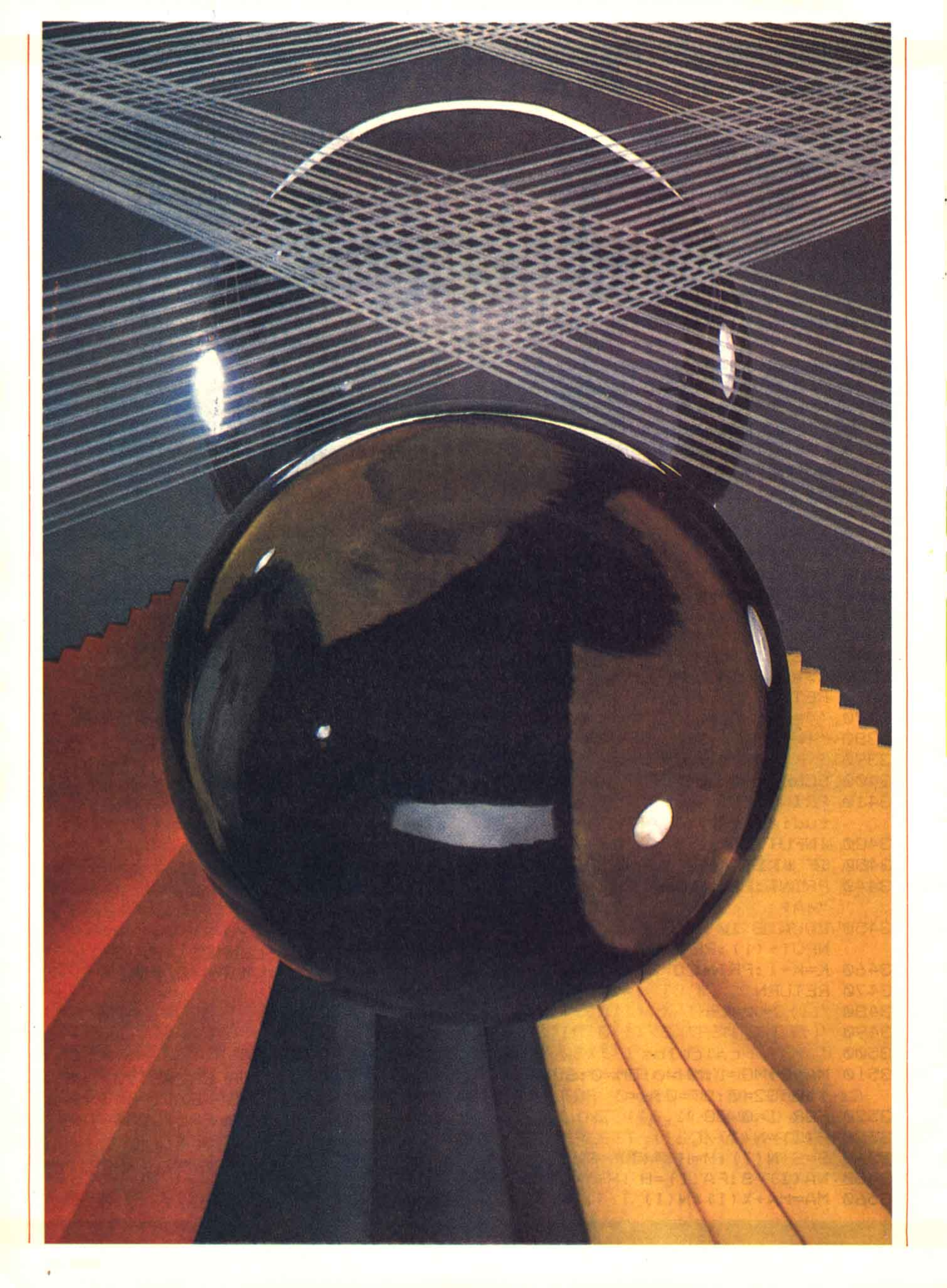
```
1300 NEXT I
1310 H=D(0):S=0
1320 FOR I=1 TO K
1330 IF D(I)>H THEN H=D(I):S=I
1340 NEXT I
1350 IF S=0 THEN H=0 ELSE H=D(S
     -1)
1360 MO=LI(S)+D(S+1)/(D(S+1)+H)
     *(LS(S)-LI(S))
1370 B$="III"
1380 '
1390 '
1400 '         **Menu II**
1410 SCREEN 2
1420 LINE(20,0)-(236,20),12,BF
1430 LINE(20,25)-(236,180),12,B
     F
1440 OPEN"GRP:"AS#1
1450 PSET(40,10),12:PRINT#1,"DI
     STRIBUCIONES TIPO ";B$
1460 RESTORE 1570
1470 FOR I=40 TO 170 STEP 17
1480 READ A$
1490 PSET(35,I),12:PRINT#1,A$
1500 NEXT I
1510 CLOSE#1
1520 A$=INPUT$(1)
1530 IF A$<"1" OR A$>"6"THEN BE
     EP:GOTO 1520
1540 IF A$="6" THEN RETURN
1550 ON VAL(A$) GOSUB 1600,2040
     ,2220,2380,2580
1560 GOTO 1410
1570 DATA 1.-TABLA DE FRECUENCI
     AS,2.-MEDIDAS DE POSICION,
     3.-MEDIDAS DE DISPERSION,4
     .-MEDIDAS DE FORMA,5.-GRAF
     ICOS,6.-REGRESO A MENU,,"
              Pulse Opción ?"
1580 '
1590 '
1600 ' tabla de frecuencias
1610 SCREEN 0:WIDTH 40
1620 IF B$="III"THEN 1760
1630 PRINT TAB(15) "FRECUENCIAS
     "
1640 PRINT TAB(5);"Xi";TAB(12);
     "ni";TAB(19);"Ni";TAB(27);
     "fi";TAB(34);"Fi"
1650 PRINT "    -------  ----  --
     ----  -----  -----"
1660 FOR I=0 TO K
1670 J=I+3
1680 LOCATE 2,J:PRINT USING "##
     ##.##";X(I)
1690 LOCATE11,J:PRINT USING "##
     ##";N(I)
1700 LOCATE17,J:PRINT USING "##
     ####";NA(I)
1710 LOCATE25,J:PRINT USING "#.
     ###";F(I)
1720 LOCATE32,J:PRINT USING "#.
     ###";FA(I)
1730 NEXT I
1740 LOCATE 8,24:PRINT T$;
1750 A$=INPUT$(1):RETURN
1760 PRINT TAB(9);"FRECUENCIAS
     ABSOLUTAS"
1770 PRINT TAB(5);"li";TAB(13);
     "Li";TAB(21);"Xi";TAB(27);
     "ni";TAB(33);"Ni"
1780 PRINT "    ------- ------- -
     ------ ---- ------"
1790 FOR I=0 TO K
1800 J=I+3
1810 LOCATE 2,J:PRINT USING "##
     ##.##";LI(I)
1820 LOCATE10,J:PRINT USING "##
     ##.##";LS(I)
1830 LOCATE18,J:PRINT USING "##
     ##.##";X(I)
1840 LOCATE26,J:PRINT USING "##
     ##";N(I)
1850 LOCATE31,J:PRINT USING "##
     ####";NA(I)
1860 NEXT I
1870 LOCATE 8,24:PRINT T$;
1880 A$=INPUT$(1):CLS
1890 PRINT SPC(9);"FRECUENCIAS
     RELATIVAS"
1900 PRINT TAB(5);"li";TAB(13);
     "Li";TAB(21);"Xi";TAB(27);
     "fi";TAB(33);"Fi"
1910 PRINT "    ------- ------- -
     ------ ---- -----"
1920 FOR I=0 TO K
1930 J=I+3
1940 LOCATE 2,J:PRINT USING "##
     ##.##";LI(I)
1950 LOCATE10,J:PRINT USING "##
     ##.##";LS(I)
1960 LOCATE18,J:PRINT USING "##
     ##.##";X(I)
```

```
1970 LOCATE26,J:PRINT USING "#.
     ###";F(I)
1980 LOCATE32,J:PRINT USING "#.
     ###";FA(I)
1990 NEXT I
2000 LOCATE 8,24:PRINT T$;
2010 A$=INPUT$(1):RETURN
2020 '
2030 '
2040 '  medidas de posición
2050 SCREEN 0:WIDTH 32
2060 PRINT TAB(6);"MEDIDAS DE P
     OSICION"
2070 PRINT D$:PRINT"MEDIAS"
2080 PRINT USING "  Aritmética.
     ...:#.######^^^^";MA
2090 IF W1=1 THEN 2110
2100 PRINT USING "  Geométrica.
     ...:#.######^^^^";MG
2110 IF W2=1 THEN 2130
2120 PRINT USING "  Armónica...
     ...:#.######^^^^";MH
2130 PRINT D$
2140 PRINT USING "  Mediana....
     ...:#.######^^^^";ME
2150 PRINT USING "  Moda.......
     ...:#.######^^^^";MO
2160 PRINT USING "  Valor Máxim
     o..:#.######^^^^";X(K)
2170 PRINT USING "  Valor Mínim
     o..:#.######^^^^";X(0)
2180 PRINT D$:PRINT TAB(5);T$
2190 A$=INPUT$(1):RETURN
2200 '
2210 '
2220 '  medidas de dispersion
2230 SCREEN 0:WIDTH 32
2240 PRINT TAB(6);"MEDIDAS DE D
     ISPERSION"
2250 PRINT D$:PRINT "ABSOLUTAS"
2260 PRINT USING "  Rango......
     ..:#.######^^^^";RE
2270 PRINT USING "  Des. a. med
     ia:#.######^^^^";DM
2280 PRINT USING "  Varianza...
     ..:#.######^^^^";S2
2290 PRINT USING "  Desv. tipic
     a.:#.######^^^^";DT
```


GANE
7.000 PTAS.
todos los meses
PARTICIPANDO
EN NUESTRO
CONCURSO
MSX Magazine premiará cada mes los programas que nos hagan llegar nuestros lectores.
Para participar en este concurso abierto, todo aficionado a los ordenadores con este estándar deberá hacer llegar a la redacción de la revista el listado, un cassette y un texto explicativo.
Entre todos los programas que recibamos cada mes, serán seleccionados para su publicación aquellos que reúnan los siguientes criterios:
— Originalidad de la aplicación.
— Simplicidad del método de programación.
La única condición para participar en el concurso será que los programas no hayan sido publicados previamente en ninguna revista.
Enviar vuestros programas a: MSX Magazine
C/Bravo Murillo, 377 - 5.° A 28020 MADRID
BANCO DE ESPAÑA
2E1554046
MIL pesetas
1000
CINCO MIL pesetas
5000
14457150

```
2300 PRINT D$:PRINT "RELATIVAS"
2310 PRINT USING "  C. apertura
     ..:#.######^^^^";CA
2320 PRINT USING "  R. relativo
     ..:#.######^^^^";RR
2330 PRINT USING "  c. variacio
     n.:#.######^^^^";CV
2340 PRINT D$:PRINT TAB(5);T$
2350 A$=INPUT$(1):RETURN
2360 '
2370 '
2380 '  medidas de forma
2390 SCREEN 0:WIDTH 32
2400 PRINT TAB(8);"MEDIDAS DE F
     ORMA"
2410 PRINT D$:PRINT"ASIMETRIA"
2420 PRINT USING" C. Fisher....
     :#.######^^^^";G1
2430 PRINT
2440 IF G1<0 THEN PRINT TAB(4);
      "Asimétrica a izquierda"
2450 IF G1=0 THEN PRINT TAB(11)
     ;"Simétrica"
2460 IF G1>0 THEN PRINT TAB(4);
      "Asimétrica a derecha"
2470 PRINT:PRINT D$:PRINT "CURT
     OSIS"
2480 PRINT USING" C. Curtosis..
     :#.######^^^^";G2
2490 PRINT
2500 IF G2<0 THEN PRINT TAB(3);
      "Distribución platicurtic
     a"
2510 IF G2=0 THEN PRINT TAB(6);
      "Distribución normal"
2520 IF G2>0 THEN PRINT TAB(3);
      "Distribución leptocúrtic
     a"
2530 PRINT:PRINT D$:PRINT TAB(5
     );T$
2540 A$=INPUT$(1):RETURN
2550 '
2560 '
2570 '  gràficos
2580 CLS:OPEN"grp:"AS#1
2590 IF B$="III" THEN 3000
2600 A$="fi":C$="DIAGRAMA DE BA
     RRAS"
2610 GOSUB 3840
2620 X=F(0):S=F(0)
2630 FOR I=1 TO K
2640 IF F(I)>X THEN X=F(I)
2650 IF F(I)<S THEN S=F(I)
2660 NEXT I
2670 H1=(X-S)/10:Y=S
2680 FOR I=144 TO 24 STEP -12
2690 LINE (46,I)-(48,I)
2700 PRESET (16,I-4):PRINT #1,U
     SING "#.##";Y
2710 Y=Y+H1
2720 NEXT I
2730 P1=120/(S-X):H1=24-P1*X
2740 FOR I=0 TO K
2750 X=X(I)*P+H:Y=F(I)*P1+H1
2760 LINE (X,160)-(X,Y)
2770 NEXT I
2780 PRESET (40,182):PRINT#1,T$
2790 C$=INPUT$(1):CLS
2800 A$="Fi":C$="DIAGRAMA ACUMU
     LATIVO"
2810 GOSUB 3840
2820 J=0
2830 FOR I=160 TO 20 STEP -14
2840 LINE (46,I)-(48,I)
2850 PRESET (24,I-4):PRINT #1,U
     SING "#.#";J
2860 J=J+.1
2870 NEXT I
2880 X=X(0)*P+H
2890 PSET (X,160)
2900 FOR I=0 TO K-1
2910 X=X(I)*P+H:S=X(I+1)*P+H
2920 Y=160-FA(I)*140
2930 LINE -(X,Y):LINE -(S,Y)
2940 NEXT I
2950 Y=160-FA(K)*140
2960 LINE -(S,Y):LINE -(S+4,Y)
2970 PRESET(40,182):PRINT#1,T$
2980 C$=INPUT$(1):CLOSE#1
2990 RETURN
3000 A$="di":C$="HISTOGRAMA"
3010 GOSUB 3840
3020 X=D(0):S=D(0)
3030 FOR I=1 TO K
3040 IF D(I)>X THEN X=D(I)
3050 IF D(I)<S THEN S=D(I)
3060 NEXT I
3070 H1=(X-S)/10:Y=S
3080 FOR I=144 TO 24 STEP -12
3090 LINE (46,I)-(48,I)
3100 PRESET (16,I-4):PRINT#1,US
     ING "####";Y
3110 Y=Y+H1
3120 NEXT I
```

```
3130 P1=120/(S-X):H1=24-P1*X
3140 FOR I=0 TO K
3150 X=LI(I)*P+H:S=LS(I)*P+H:Y=
     D(I)*P1+H1
3160 LINE(X,160)-(S,Y),,B
3170 NEXT I
3180 PRESET (40,182):PRINT#1,T$
3190 C$=INPUT$(1):CLS
3200 A$="Fi":C$="POLIGONO ACUMU
     LATIVO"
3210 GOSUB 3840
3220 J=0
3230 FOR I=160 TO 24 STEP -13.6
3240 LINE (46,I)-(48,I)
3250 PRESET(24,I-4):PRINT#1,USI
     NG "#.#";J
3260 J=J+.1
3270 NEXT I
3280 X=X(0)*P+H:Y=160-136*FA(0)
3290 PSET(X,Y)
3300 FOR I=1 TO K
3310 X=X(I)*P+H:Y=160-136*FA(I)
3320 LINE -(X,Y)
3330 NEXT I
3340 PRESET(40,182):PRINT#1,T$
3350 C$=INPUT$(1):CLOSE#1
3360 RETURN
3370 '
3380 '
3390 '    **Subrutinas**
3400 SCREEN 0:WIDTH 32
3410 PRINT "Número de ";A$;" es
     tudiadas"
3420 INPUT K
3430 IF K<21 THEN 3460
3440 PRINT:PRINT"solo hasta 20
     ";A$
3450 LOCATE 10,23:PRINT T$:A$=I
     NPUT$(1):RETURN
3460 K=K-1:PRINT
3470 RETURN
3480 '
3490 '
3500 '        cálculos
3510 MA=0:MG=1:MH=0:DM=0:S2=0:G
     1=0:G2=0:DT=0:H=0
3520 FOR I=0 TO K
3530 F(I)=N(I)/C
3540 S=S+N(I):H=H+F(I)
3550 NA(I)=S:FA(I)=H
3560 MA=MA+X(I)*N(I)
3570 MG=MG*X(I)^N(I)
3580 MH=MH+N(I)/X(I)
3590 NEXT I
3600 MA=MA/C' m.aritmética
3610 IF W1=1 THEN 3630
3620 MG=MG^(1/C)' m.gométrica
3630 IF W2=1 THEN 3650
3640 MH=C/MH' m.armónica
3650 RE=X(K)-X(0)' recorrido
3660 CA=X(K)/X(0)' c.apertura
3670 RR=RE/MA' r.relativo
3680 FOR I=0 TO K
3690 H=X(I)-MA
3700 DM=DM+ABS(H)*N(I)
3710 S2=S2+H^2*N(I)
3720 G1=G1+H^3*N(I)
3730 G2=G2+H^4*N(I)
3740 NEXT I
3750 DM=DM/C' d.a.media
3760 S2=S2/C' varianza
3770 DT=SQR(S2)' d.típica
3780 CV=DT/MA' c.variación
3790 G1=(G1/C)/S^3' asimetria
3800 G2=(G2/C)/S^4-3' curtosis
3810 RETURN
3820 '
3830 '
3840 '       gráfica
3850 LINE(48,160)-(240,0),4,BF
3860 PRESET (230,172):PRINT#1,"
     Xi"
3870 IF B$="II" THEN X=X(0):Y=X
     (K) ELSE X=LI(0):Y=LS(K)
3880 P=176/(Y-X):H=56-P*X
3890 FOR I=0 TO K
3900 X=X(I)*P+H:LINE(X,160)-(X,
     162)
3910 NEXT I
3920 PRESET(24,4):PRINT#1,A$
3930 X=INT(256-LEN(C$)*8)/2+16
3940 PSET (X,4),4:PRINT#1,C$
3950 RETURN
3960 '
3970 '
3980 '      errores
3990 IF ERL=420 OR ERL=3570 THE
     N W1=1:RESUME NEXT
4000 IF ERL=430 OR ERL=3580 THE
     N W2=1:RESUME NEXT
4010 CLS:PRINT"error en linea "
     ;ERL
4020 END
```



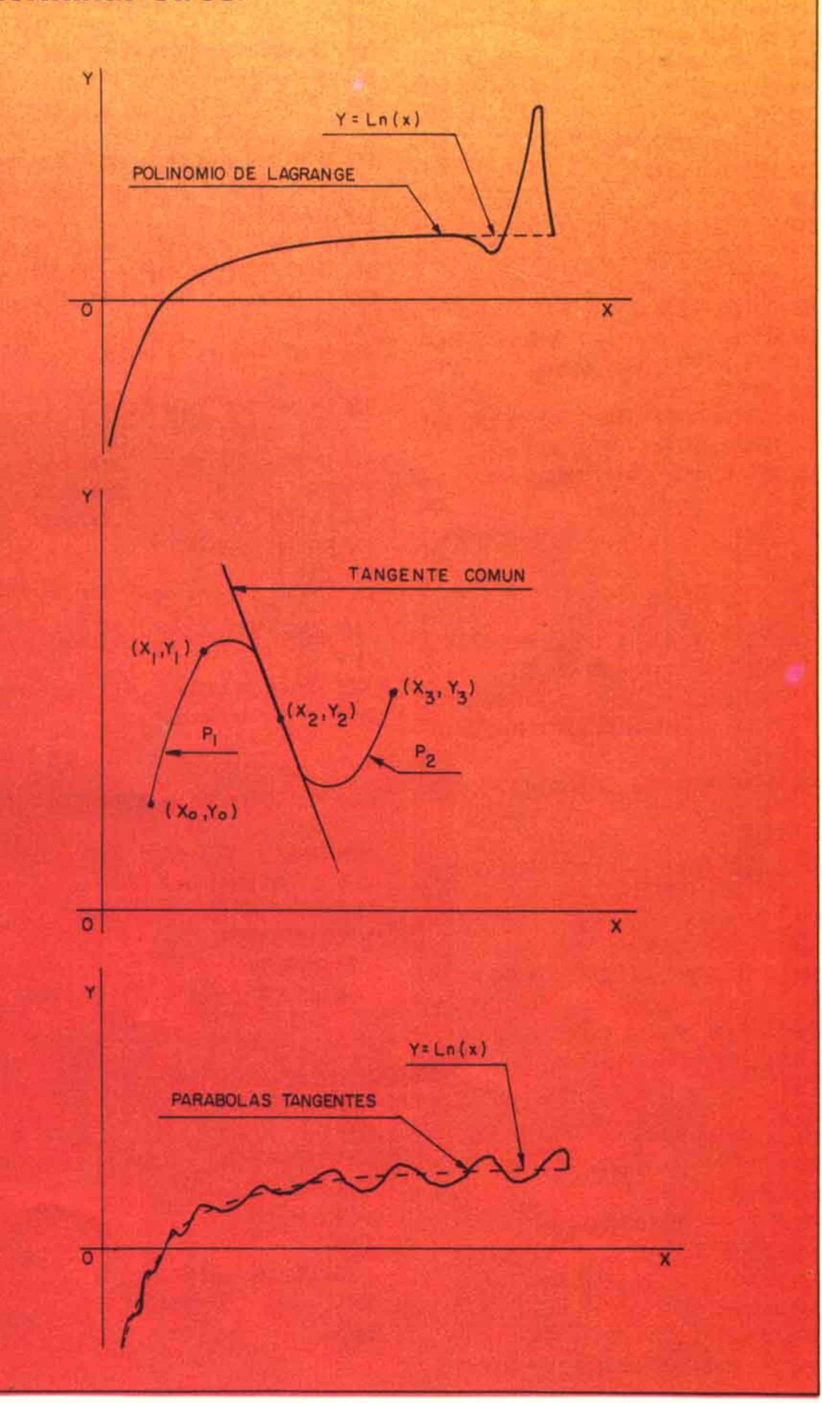
Interpolación

En Matemática Aplicada se presenta con cierta frecuencia el caso de conocer de una función sólo un número determinado de puntos. Se nos plantea entonces la necesidad de, conocido ese número finito de puntos, determinar otros.

Existen varios métodos para ello, llamados Métodos de Interpolación, que consisten en encontrar una función que pase por todos los puntos que conocemos. Como es lógico, estos métodos no son exactos, pero permiten tener una idea bastante aproximada del comportamiento de la función desconocida. Una vez más, el ordenador se convierte en una herramienta tremendamente útil, ya que cuando el número de puntos conocidos es superior a cuatro o cinco, resulta francamente tedioso encontrar esa función de interpolación.

Vamos a conocer dos métodos de interpolación. El primero de ellos consistirá en encontrar un polinomio que pase por todos los puntos conocidos, y el segundo en encontrar una parábola que, siendo de eje paralelo al de ordenadas, pase por dos puntos adyacentes, siendo tangente a la parábola anterior en el primero de esos puntos. Como es lógico, cuanto mayor sea el número de puntos conocidos, más se acercará la función obtenida a la desconocida.

Como ejemplo, en ambos casos, estudiaremos la función correspondiente que pase pqr los siguientes puntos (que pertenecen a la función Y=ln(x)):



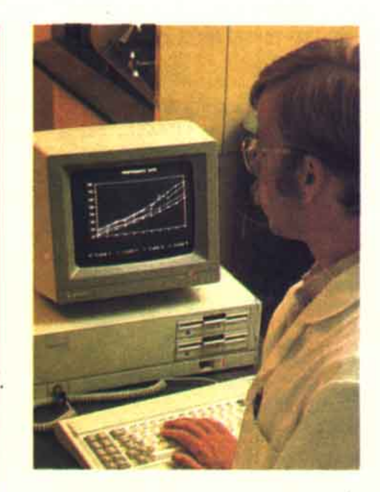
```
10 'METODOS DE INTERPOLACION
20 '========================
30 '
40 'Juan Antonio Feberero Castejón
50 'Versión 07.02.04.86 - 6686 Bytes
60 '
70 '
80 'Inicialización
90 '--------------
100 '
110 CLEAR 200,46999!
120 SCREEN 0,,0
130 WIDTH 39
140 KEY OFF
150 FOR I=5 TO 10
160 KEY I,""
170 NEXT I
180 GOSUB 3470
190 ON ERROR GOTO 3370
200 ON KEY GOSUB 3160,3180,3200,3250
210 KEY(1) ON
220 KEY(2) ON
230 ON STOP GOSUB 3400
240 STOP ON
250 '
260 'Rutinas C.M.
270 '------------
280 '
290 FOR I=1 TO 50
300 READ W$
310 POKE &HEFFF+I,VAL("&H"+W$)
320 NEXT I
330 DEFUSR=&HF000 'VRAM a RAM
340 DEFUSR1=&HF019 'RAM a VRAM
350 DEFUSR2=&H3E 'Rest. teclas func.
360 '
370 'Menu
380 '====
390 '
400 LOCATE 6,5
410 PRINT "POLINOMIOS de INTERPOLACION"
420 PRINT TAB(5);STRING$(29,195)
430 LOCATE 6,9
440 PRINT "1. Método de Lagrange."
450 PRINT
460 PRINT TAB(6);"2. Interpolación por p
    arábolas."
470 PRINT
480 PRINT TAB(6);"3. Interpolación por r
    ectas."
490 WM$=INKEY$
500 IF WM$="" THEN 490
510 IF INSTR("123",WM$)=0 THEN 490
520 '
530 'Entrada de datos
540 '================
550 '
560 CLS
570 LOCATE 10,10
580 PRINT "1. Datos de teclado."
590 LOCATE 10,12
600 PRINT "2. Datos de cinta."
610 LOCATE 10,14
620 PRINT "3. Datos ya en memoria."
630 WD$=INKEY$
640 IF WD$="" THEN 630
650 IF INSTR("123",WD$)=0 THEN 630
660 CLS
670 ON VAL(WD$) GOTO 720,1050,790
680 '
690 'Datos de teclado
700 '----------------
710 '
720 INPUT "Número de datos";N
730 ERASE X,Y,P,A,B,C
740 DIM X(N),Y(N),P(N),A(N-3),B(N-3),C(N
    -3)
750 FOR I=1 TO N
760 PRINT USING "X(##),y(##)";I;I;
770 INPUT X(I),Y(I)
780 NEXT I
790 GOSUB 1230
800 CLS
810 '
820 'Grabación de datos
830 '------------------
840 '
850 PRINT "¿Deseas grabar los datos en c
    inta? S?N"
860 W1$=INKEY$
870 IF W1$="" THEN 860
880 IF INSTR("SNsn",W1$)=0 THEN 860
890 IF INSTR("Ss",W1$)=0 THEN 1460
900 PRINT
910 PRINT "Prepara la grabadora... [RETU
    RN]."
920 IF INKEY$="" THEN 920
930 OPEN "INTERP" FOR OUTPUT AS#1
940 PRINT#1, STR$(N)    'Datos como
950 FOR I=1 TO N        'cadenas de
960 PRINT#1, STR$(X(I)) 'caracteres
970 PRINT#1, STR$(Y(I))
980 NEXT I
990 CLOSE #1
1000 GOTO 1460
1010 '
1020 'Datos de cinta
1030 '--------------
1040 '
1050 PRINT "Prepara la grabadora... [RET
     URN]."
1060 ERASE X,Y,P,A,B,C
1070 IF INKEY$="" THEN 1070
1080 OPEN "INTERP" FOR INPUT AS#1
1090 LINE INPUT#1, A$
1100 N=VAL(A$)
1110 DIM X(N),Y(N),P(N),A(N-3),B(N-3),C(
     N-3)
1120 FOR I=1 TO N
1130 LINE INPUT#1, A$
1140 X(I)=VAL(A$)
1150 LINE INPUT#1, A$
1160 Y(I)=VAL(A$)
1170 NEXT I
1180 CLOSE #1
1190 '
1200 'Listado de datos
1210 '----------------
1220 '
1230 CLS
1240 FOR I=1 TO N
1250 KEY(4) STOP
1260 PRINT USING "X,Y(##)=&";I;STR$(X(I)
     );
1270 LOCATE 22
1280 PRINT ";";Y(I)
1290 IF I=N THEN 1310
1300 IF CSRLIN<20 THEN 1410
1310 KEY(4) ON
1320 W$=INKEY$
1330 IF W$="" THEN 1320
1340 IF W$=CHR$(30) THEN I=I-CSRLIN-20:G
     OTO 1380
1350 IF W$=CHR$(31) THEN 1390
1360 IF W$=CHR$(32) THEN IF WD$="1" THEN
      RETURN ELSE 1460
1370 GOTO 1320
1380 IF I<0 THEN I=0
1390 IF I=N THEN 1320
1400 CLS
1410 NEXT I
1420 '
1430 'Cálculos iniciales
1440 '------------------
1450 '
1460 CLS
1470 LOCATE 8,11
1480 PRINT "Un momento, por favor"
1490 ON VAL(WM$) GOSUB 2000,2310,2760
1500 '
1510 'Gráfica
1520 '-------
1530 '
1540 CLS
1550 PRINT "¿Deseas salida gráfica? S/N"
1560 W$=INKEY$
1570 IF W$="" THEN 1560
1580 IF INSTR("SNsn",W$)=0 THEN 1560
1590 IF INSTR("Ss",W$)=0 THEN 1860
1600 KEY(2) STOP
1610 GOSUB 2950
1620 IF XN<>0 THEN XI=0:GOTO 1650
1630 XI=127
1640 LINE (0,0)-(127,0),10
1650 XE=XI
1660 FOR X1=XI TO 250
1670 BEEP
1680 PSET (X1,0),10
1690 X=(X1-127)*XS/125
1700 ON VAL(WM$) GOSUB 2100,2660,2830
1710 IF FLAG=1 THEN FLAG=0:XE=X1+1:GOTO
     1760
1720 Y=95-90*Y/XS
1730 IF X1=XE THEN PSET(X1,Y) ELSE LINE
     (XA,YA)-(X1,Y)
1740 XA=X1
1750 YA=Y
1760 NEXT X1
1770 LINE (0,0)-(255,0),4
1780 U=USR(0)
1790 KEY(2) ON
1800 KEY(3) ON
1810 BEEP:IF INKEY$="" THEN 1810
1820 '
1830 'Interpolación
1840 '-------------
1850 '
```

```
1860 INPUT "X";X
1870 ON VAL(WM$) GOSUB 2100,2660,2830
1880 IF FLAG=1 THEN PRINT "X fuera de límites":PRINT:FLAG=0:GOTO 1860
1890 PRINT "Y=";Y
1900 PRINT
1910 GOTO 1860
1920 '
1930 'Método de Lagrange
1940 '==================
1950 '
1960 '
1970 'Cálculo de polinomios
1980 '---------------------
1990 '
2000 FOR I=1 TO N
2010 X=X(I)
2020 GOSUB 2200
2030 P(I)=Y(I)/PX
2040 NEXT I
2050 RETURN
2060 '
2070 'Interpolación
2080 '-------------
2090 '
2100 Y=0
2110 FOR I=1 TO N
2120 GOSUB 2200
2130 Y=Y+P(I)*PX
2140 NEXT I
2150 RETURN
2160 '
2170 'Rutina polinomios
2180 '-----------------
2190 '
2200 IF X<X(1) OR X>X(N) THEN FLAG=1:RETURN
2210 PX=1
2220 FOR J=1 TO N
2230 IF J=I THEN 2250
2240 PX=PX*(X-X(J))
2250 NEXT J
2260 RETURN
2270 '
2280 'Interpolación por parábolas
2290 '===========================
2300 '
2310 DEFFNY(A,B,C)=X(A)*(Y(B)-Y(C))
2320 DEFFNX(A,B,C)=X(A)*(X(B)^2-X(C)^2)
2330 DEFFNC(A,B)=Y(A)-X(A)^2*A(B)-X(A)*B(B)
2340 '
2350 'Ordenación
2360 '----------
2370 '
2380 I=N
2390 I=INT(I/2):IF I<1 THEN 2470
2400 N1=1:K=N-I
2410 M=N1
2420 H=M+I
2430 IF X(M)<X(H) THEN 2460
2440 SWAP X(M),X(H):SWAP Y(M),Y(H)
2450 M=M-I:IF M<1 THEN 2460 ELSE 2420
2460 N1=N1+1:IF N1>K THEN 2390 ELSE 2410
2470 IF WM$="3" OR F1=1 THEN RETURN
2480 '
2490 'Cálculo de coeficientes
2500 '-----------------------
2510 '
2520 A(0)=(FNY(1,2,3)+FNY(2,3,1)+FNY(3,1,2))/(FNX(1,2,3)+FNX(2,3,1)+FNX(3,1,2))
2530 B(0)=(Y(2)-Y(1)-(X(2)^2-X(1)^2)*A(0))/(X(2)-X(1))
2540 C(0)=FNC(1,0)
2550 FOR I=4 TO N
2560 AB=2*A(I-4)*X(I-1)+B(I-4)
2570 A(I-3)=(Y(I)-Y(I-1)-(X(I)-X(I-1))*AB)/(X(I)-X(I-1))^2
2580 B(I-3)=AB-2*X(I-1)*A(I-3)
2590 C(I-3)=FNC(I-1,I-3)
2600 NEXT I
2610 IF WM$="3" THEN RETURN
2620 '
2630 'Interpolación
2640 '-------------
2650 '
2660 IF X<X(1) OR X>X(N) THEN FLAG=1:RETURN
2670 FOR I=3 TO N
2680 IF X<=X(I) THEN I=I-3:GOTO 2700
2690 NEXT I
2700 Y=A(I)*X^2+B(I)*X+C(I)
2710 RETURN
2720 '
2730 'Interpolación por rectas.
2740 '========================
2750 '
2760 DEFFNYR(I,X)=(Y(I)-Y(I-1))*(X-X(I-1))/(X(I)-X(I-1))+Y(I-1)
2770 GOSUB 2380
2780 RETURN
2790 '
2800 'Interpolación
2810 '-------------
2820 '
2830 IF X<X(1) OR X>X(N) THEN FLAG=1:RETURN
2840 FOR I=2 TO N
2850 IF X<=X(I) THEN 2870
2860 NEXT I
2870 Y=FNYR(I,X)
2880 RETURN
2890 '
2900 'Preparación pantalla gráfica
2910 '----------------------------
2920 '
2930 F1=1
2940 GOSUB 2380
2950 XP=X(N)
2960 XN=X(1)
2970 IF XN>0 THEN XN=0
2980 YP=0
2990 YN=0
3000 FOR IS=1 TO N
3010 IF Y(IS)>YP THEN YP=Y(IS)
3020 IF Y(IS)<YN THEN YN=Y(IS)
3030 NEXT IS
3040 IF ABS(XP)>ABS(XN) THEN XS=ABS(XP) ELSE XS=ABS(XN)
3050 IF ABS(YP)>ABS(YN) THEN YS=ABS(YP) ELSE YS=ABS(YN)
3060 IF YS>XS THEN XS=YS
3070 COLOR 15,4,4
3080 SCREEN 2
3090 LINE (127,0)-(127,191)
3100 LINE (0,95)-(255,95)
3110 RETURN
3120 '
3130 'Rutinas On key gosub
3140 '--------------------
3150 '
3160 MOTOR      'Key 1:Motor
3170 RETURN     '-----------
3180 SCREEN 0   'Key 2:Menú
3190 RETURN 400 '----------
3200 SCREEN 2   'Key 3:Gráfica
3210 U=USR1(0)  '-------------
3215 KEY (3)ON
3220 IF INKEY$="" THEN 3220
3230 SCREEN 0
3240 RETURN
3250 LOCATE 0,21'Key 4:Modif. datos
3260            '------------------
3270 INPUT "Indice del dato a modificar";IND
3280 PRINT USING "X(##),Y(##)";IND;IND;
3290 INPUT X(IND),Y(IND)
3300 CLS
3310 IF I=N THEN I=0
3320 RETURN 1380
3330 '
3340 'Errores
3350 '-------
3360 '
3370 IF ERR=5 AND ERL=730 OR ERL=1060 THEN RESUME NEXT
3380 IF ERR=6 THEN RESUME NEXT
3390 PRINT "Error";ERR;" en linea";ERL
3400 U=USR2(0) 'On stop...Rest teclas
3410 KEY ON    '--------------------
3420 STOP
3430 '
3440 'Teclas de función
3450 '-----------------
3460 '
3470 KEY 1,"Motor"
3480 KEY 2,"Menú"
3490 KEY 3,"Gráfica"
3500 KEY 4,"Modific"
3510 KEY ON
3520 RETURN
3530 '
3540 'Datas C.M.
3550 '----------
3560 '
3570 DATA 21,00,00,01,00,18,11,98,B7,CD,59,00,21,00,20,01,00,18,11,9E,CF,CD,59,00,C9 'VRAM a RAM
3580 DATA 21,98,B7,01,00,18,11,00,00,CD,5C,00,21,9E,CF,01,00,18,11,00,20,CD,5C,00,C9 'RAM a VRAM
```

x	Ø,1	Ø,3	Ø,6	Ø,9	1,Ø	1,2	1,5	2,Ø	2,5
y=f(x)	-2,3Ø26	-1,2Ø4Ø	-Ø,51Ø8	-Ø,1Ø54	Ø,Ø	Ø,1823	Ø,4Ø55	Ø,6931	Ø,9163

x	3,Ø	3,5	4,Ø	4,5	5,Ø	5,5	6,Ø	6,5
y=f(x)	1,Ø986	1,2528	1,3863	1,5Ø41	1,6Ø94	1,7Ø47	1,7918	1,8718

hallaremos los valores de y correspondientes a los siguientes valores de x: 0,5,, 0,95,, 1,3,, 1,85,, 5,55,, 6,3.



El método de lagrange falla frecuentemente en los extremos del intervalo.

Método de Lagrange

Siempre que una función sea contínua y derivable en un punto, es posible encontrar una serie polinómica ilimitada que aproxime a la función en dicho punto. De hecho, las funciones incorporadas en las calculadoras y los ordenadores (sen(x), tg(x), exp(x), log(x)) se calculan mediante una serie polinómica (Desarrollos en serie de Taylor-Mc Laurin), por ejemplo, para sen(x) exp(x) tenemos:

$$\text{sen}(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \ldots$$

$$\exp(x) = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \ldots$$

Sí, en general, tenemos *n* puntos de una función $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$, siempre podemos encontrar un polinomio:

$$y = a_o + a_1 x + a_2 x^2 + a_o + \ldots + a_n x^n$$

que pase por todos ellos.

Lagrange consideró y como suma de *n* monomios de la forma:

$$y = \sum_{i=1}^{n} M_i(x) \qquad (2)$$

de tal forma que el monomio i se hace igual a y_i para $x=x_i$, y se anula para el resto de los puntos conocidos $(x_1, y_1), (x_2, y_2) \ldots (x_{i-1}, y_{i-1}), (x_{i+1}, y_{i+1}), \ldots (x_n, x_n)$:

$$M_i(x_i) = y_i$$
$$M_i(x_j) = 0 \quad (j=1, 2, \ldots i-i, i+1, \ldots n) \quad (3)$$

Para que sea $M_i(x_i)=0$ y $M_i(x_i) \neq 0$ podemos hacer:

$M_i(x) = (x-x_1) . (x-x_2) \ldots (x-x_{i+1}) \ldots (x-x_n)$

es decir:

$$M_i(x) = \prod_{\substack{j \neq i \\ j=1}}^{n} (x - x_j)$$

De esta forma cada factor $(x-x_j)$ se anula $(x_j - x_j = 0)$ para cada $x=x_j$ y no lo hace para $x=x_i$. Si ahora dividimos (4) por:

$(x_i - x_1).(x_i - x_2) \ldots (x_i - x_{i-1}).(x_i - x_{i+1}) \ldots (x_i - x_n)$

es decir, haciendo:

$$M_i(x) = \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)} \qquad (5)$$

$M_i(x_j)$ seguirá siendo cero y ahora $M_i(x_i)$ será igual a 1. Por tanto basta multiplicar (5) por y para que se cumplan las condiciones (3). Definitivamente será:

$$y(x) = \sum_{i=1}^{n} Y_i \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)}$$

que es la *Fórmula de Lagrange.*

Construyamos ahora un algoritmo para calcular y(x):

1. Calculamos los productos:

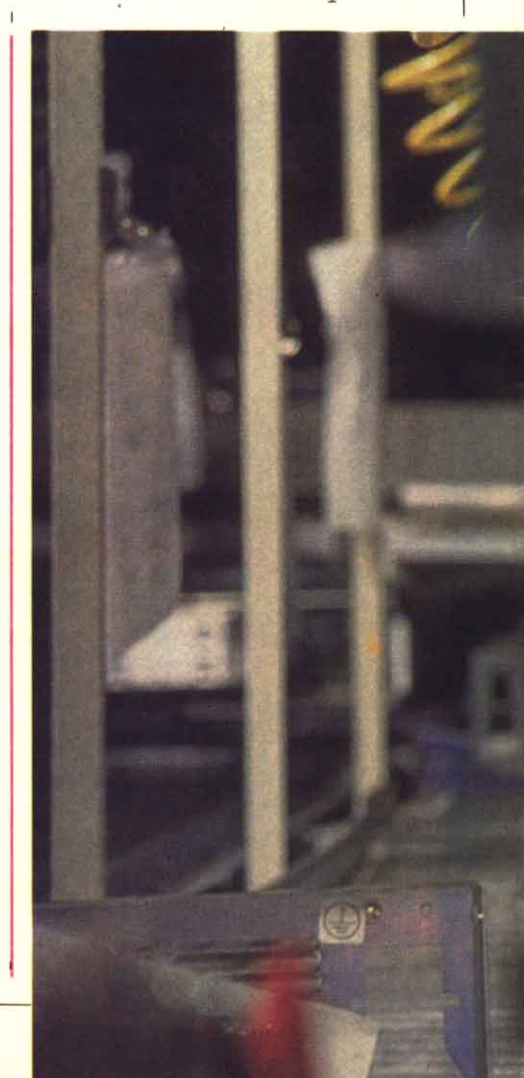
$$P_i = \prod_{j \neq i} (x_i - x_j)$$

2. Hacemos:

$$A_i = \frac{Y_i}{P_i}$$

3. Para cada valor de x calculamos los productos:

$$Q_i = \prod_{j \neq i} (x - x_i)$$



4. Hallamos la suma:

$$y(x) = \sum_{i=1}^{n} A_i Q_i$$

Utilizando este algoritmo se ha preparado la rutina correspondiente del programa listado al final. En la figura 1 hemos dibujado la función y=1n(x) y el polinomio de interpolación de *Lagrange* correspondiente a los valores enunciados al principio de este artículo.

Interpolación por parábolas tangentes

El método de *Lagrange*, además de ser lento falla frecuentemente en los extremos del intervalo, dando unos errores bastante importantes; así, en el ejemplo que hemos tomado da un valor de y=3.0519 para x=6.3, cuando el valor exacto de la función y=1n(x) para x=6.3 es y=1.8405.

Si en lugar de un polinomio de grado *n* tomamos parábolas que sean tangentes entre sí y que pasen por todos los puntos dados, tendremos (según los casos) una curva que representa la función que estamos interpolando peor que el método de *Lagrange*, pero los errores obtenidos en los extremos del intervalo serán mucho menores. (Para el valor de x anterior, el método que vamos a aplicar de y=1.9825).

Si por (x_o, y_o), (x_1, y_1) y (x_2, y_2) hacemos pasar una parábola (P_1) de eje paralelo a OY, y por (x_2, y_2) y (x_3, y_3) hacemos pasar otra (P_2) de eje también paralelo a OY, y tal que las tangentes a (P_1) y a (P_2) por (x_2, y_2) sean coincidentes, y luego hacemos lo mismo con (x_3, y_3) y (x_4, y_4), y así sucesivamente, habremos obtenido una curva contínua con derivada contínua que pase por todos los puntos. (Algo parecido hizo *Simpson* para encontrar su célebre fórmula de integración.

Así tenemos que para un valor de x comprendido entre x_o y x_2, el valor de y vendrá dado por la parábola:

$$y = A_1x^2 + B_1x + C_1 \quad (7)$$

en el que A_1, B_1 y C_1 viene dados como solución del sistema de ecuaciones resultante de hacer que los puntos (x_o, y_o), (x_1, y_1) y (x_2, y_2) pertenezcan a la parábola (7), y valen:

$$A_1 = \frac{x_1(y_2-y_3) + x_2(y_3-y_1) + x_3(y_1-y_2)}{x_1(x_2^2-x_3^2) + x_2(x_3^2-x_1^2) + x_3(x_1^2-x_2^2)}$$

$$B_1 = \frac{y_2-y_1-(x_2^2-x_1^2)\,A_1}{x_2-x_1}$$

$$C_1 = y_1-x_1^2\,A_1-x_1B_1$$

Si x está entre las abscisas x_i y x_{i+1}, la parábola correspondiente



(P_i) vendrá dada por:

$y = A_i x^2 + B_i x + C_i$

y para encontrar los valores de A_i, B_i y C_i deberemos resolver un sistema que se obtiene haciendo que los puntos (x_i, y_i) y (x_{i+1}, y_{i+1}) pertenezcan a la parábola (8) y que la tangente a dicha parábola por (x_i, x_i) coincide con la tangente a la parábola (P_{i-1}) en el mismo punto. Como resultado se tiene:

$$A_i = \frac{y_{i+1} - y_i - (x_{i+1} - x_i)(2A_{i-1}x_i + B_{i-1})}{(x_{i+1} - x_i)^2}$$

$$B_i = 2A_{i-1}x_i + B_{i-1} - 2x_i A_i$$

$$C_i = y_i - x_i^2 A_i - x_i B_i$$

Así la parábola (P_i) es tangente a la (P_{i-1}) en (x_i, y_i) y habremos conseguido una curva contínua y con derivada contínua en todos sus puntos.

Por tanto, tenemos que si $x_0 < x_2$, $y = A_1 x^2 + B_1 x + C_1$, y si $x_2 \leqslant x_i < x < x_{i+1}$, $y = A_i x^2 + B_i x + C_i$

El programa contiene una rutina que utiliza este sencillo algorirmo. Es de observar como la aplicación de este método es mucho más rápido que el de *Lagrange* y que la función, aún tomando valores menos exactos, es una curva suave y no llega a dispararse en ningún momento. En la figura 3 hemos dibujado la función de interpolación por el método que nos ocupa junto a la función y=1n(x).



Los métodos de Interpolación facilitan el conocimiento de una función desconocida.

Dejamos al lector la resolución del problema tomando como función de interpolación una línea quebrada que pase por todos los puntos.

A continuación damos los resultados de aplicar los métodos de *Lagrange* de parábolas tangentes y de línea quebrada a las abscisas propuestas al principio:

El programa que adjuntamos incluye los tres métodos que hemos comentado. En todos los casos es posible introducir los datos desde el teclado y desde cinta magnética. Una vez se han almacenado los datos en memoria podemos dibujar la función de interpolación en la pantalla gráfica de MSX.

El proceso de dibujo (sobre todo en el método de *Lagrange*) puede ser bastante lento, por lo que hemos incluido una rutina en código máquina que permite, una vez se ha dibujado la curva correspondiente almacenarla en *RAM* y obtenerla de nuevo cuantas veces queramos pulsando la tecla *F3*. Cuando el programa se detiene, bien porque esté listando los datos para su comprobación, bien porque esté en pantalla gráfica, podremos continuar nuestro trabajo pulsando la barra espaciadora. Para modificar los datos pulsaremos *F4* y *F2* nos permite volver el menú. Pulsando *F1* activamos o desactivamos el motor.

J. Antonio Feberero

y \ x	Ø,5	Ø,95	1,3	1,85	5,55	5,8	6,3
LAGRANGE	-Ø,6919	-Ø,Ø513	Ø,2624	Ø,6151	1,7ØØ4	1,6499	3,Ø519
PARABOLAS	-Ø,6145	-Ø,Ø215	Ø,3415	Ø,49Ø9	1,66Ø5	1,6159	1,9825
RECTAS	-Ø,7419	-Ø,Ø527	Ø,2567	Ø,6Ø68	1,7134	1,757Ø	1,8398
EXACTO	-Ø,6931	-Ø,Ø513	Ø,2624	Ø,6152	1,7138	1,7579	1,84Ø5

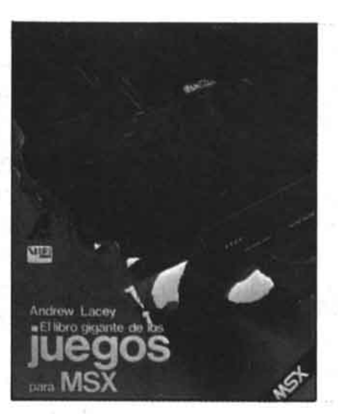


Ordena tus propias ideas

Le sacarás partido a tu ordenador



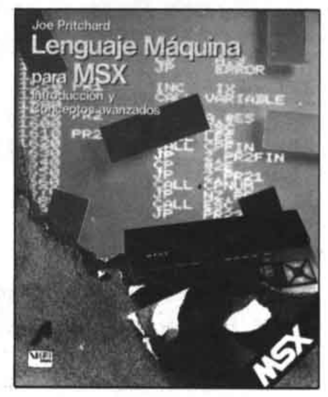
DESCUBRE TU MSX.
Programación y aplicaciones
Joe Pritchard
1.272 ptas.



EL LIBRO GIGANTE
DE LOS JUEGOS PARA MSX.
Andrew Lacey
1.590 ptas.



MSX: GUIA
DEL PROGRAMADOR
Y MANUAL DE REFERENCIA.
T. Sato, P. Mapstone e I. Muriel
2.279 ptas.



LENGUAJE MAQUINA MSX.
INTRODUCCION
Y CONCEPTOS AVANZADOS.
Joe Pritchard
1.537 ptas.



EL SUPERLIBRO DE LOS
JUEGOS PARA ORDENADOR
Tim Hartnell
2.120 ptas.



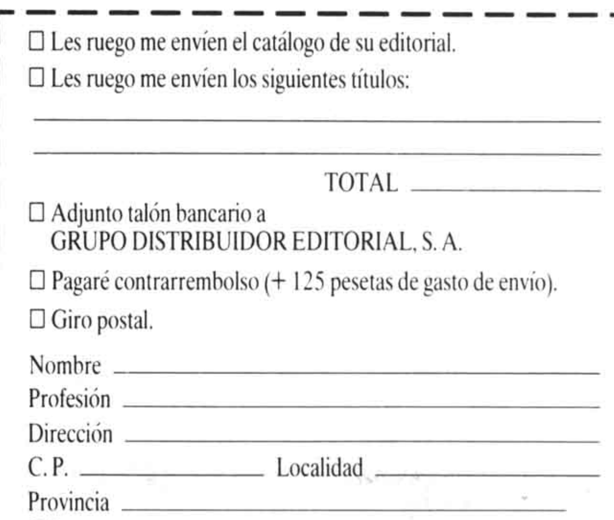
SIMULACIONES:
REPLICA LA REALIDAD
CON TU ORDENADOR
Tim Hartnell
1.643 ptas.



CODIGOS Y CLAVES
SECRETAS:
CRIPTOGRAFIA EN BASIC.
Listados para MSX, Spectrum,
Amstrad, Commodore 64, Apple II
Gareth Greenwood
1.378 ptas.



COMETAS EN TU MICRO:
EL HALLEY.
Cálculos de órbitas y parámetros
de cometas en BASIC.
F. Galende, A. Sánchez,
M. Alparaz y J. A. Sánchez García
550 ptas.


☐ Les ruego me envíen el catálogo de su editorial.
☐ Les ruego me envíen los siguientes títulos:
____
____
TOTAL __________
☐ Adjunto talón bancario a
GRUPO DISTRIBUIDOR EDITORIAL, S. A.
☐ Pagaré contrarrembolso (+ 125 pesetas de gasto de envío).
☐ Giro postal.
Nombre __________
Profesión __________
Dirección __________
C.P. __________ Localidad __________
Provincia __________
MM



Adquiéralos en su librería habitual.
Si no le es posible o desea que le enviemos nuestro catálogo, envíe este cupón a:
Apdo. de Correos 14632, Ref. D. de C. 28080 MADRID

ANAYA ANAYA ANAYA ANAYA ANAYA ANAYA

Explorador de cintas

La función de este programa es la de explorar cintas o buscar en ellas. En un principio esto es muy fácil, o por lo menos lo parece. Una posibilidad sería intentar cargar un programa



inexistente, y a partir de los mensajes *SKIP*, descubrir los que hay en la cinta. Pero ello sólo sucede cuando el programa que halla el cassette y el que buscamos, el inexistente, son del mismo tipo, por ejemplo *BLOAD*. Con este programa podemos solucionarlo.

Una vez cargado el programa y puesto en marcha, aparece en pantalla un mensaje, el cual pocos segundos después desaparecerá. Mientras el programa pone en memoria el código máquina.

A partir de entonces, entramos en el menú principal. En él tenemos tres opciones: 1) Exploración cinta, 2) Información última cinta, 3) Fin. La primera es para leer de cinta. Si entramos directamente en la segunda opción, no servirá de nada, puesto que no hemos explorado ninguna cinta.

Así pues, entramos en la primera opción, pulsando 1.

Aparece en la pantalla tres opciones más: 1) Leer cinta, la cual servirá para explorar una cinta o un trozo, hasta que se le indique por teclado el fin. Sirve, principalmente, para explorar toda una cinta. (Es posible seleccionar esta opción, e ir a dar una vuelta mientras explora la cinta). Si ya ha explorado bastante, se le indica pulsando *ctrl stop*, y se vuelve al menú principal.

En la opción 2) Buscar programa, entramos el nombre de programa a buscar, y una vez hallado para el cassette. Si sobran espacios, se rellenan con *space*.

Tanto la opción uno como la opción dos del modo exploración, al encontrar un programa, éste es escrito en la pantalla. Si se encuentra en el cassette un encabezamiento no reconocido (como fichero, prog. en error, etc.), suena un *beep*.

Una vez estamos ya en el menú principal otra vez, podemos obtener información de todos los programas hallados en la cinta pulsando la opción 2 (Información última cinta).

En ella hay cuatro opciones:

1. Listado total de los programas hallados. Para detenerlo, si ocupa más de una pantalla, pulsar *stop*. Si no se ha hallado ningún programa, o no se ha explorado ninguna cinta, aparece el mensaje: "No hay programas en memoria".

2. Información programa: tras pulsar la tecla 2, aparece el mensaje "número". Entonces debemos poner el número de orden que ocupa en la lista el programa. Entonces, si existe tal programa, obtendremos información sobre él, tal como tipo de carga *(BLOAD, LOAD, CLOAD)*, su nombre y los carácteres *ASCII* de éste. Si el programa es de tipo *BLOAD*, obtendremos además las direcciones que se encuentra, la de autoejecución y la memoria aproximada que ocupa, expresada en K.

3. Búsqueda de programas entre los hallados. Tras pulsar la tecla 3 y poner el nombre del programa, nos indica su posición en la lista, para luego pedir información o saber si se encuentra en la cinta.

4. Borrado de datos. Borra los datos de los programas hallados hasta entonces. Caben como máximo 100 programas.

Los dos principales usos de este programa son búsqueda de un programa preciso o de todo un período de cinta.

Para examinar toda una cinta se puede poner el modo uno en exploración. A partir de entonces no es preciso esperar delante del ordenador, para saber los programas que halle, puestos que en el modo información podrá conocerlo todo sobre ellos.

Jaume María Saguer
Gerona

```
10 '
20 ' Explorador de cintas. 1985
30 '
40 CLEAR1000,&HD9FF
50 ONSTOPGOSUB1340:STOP ON
60 GOSUB1240
70 GOSUB970
80 NO=3:GOSUB1050
90 ONOPGOSUB120,400
100 IFOP-3GOTO70
110 KEY ON:CLS:END
120 'Explora
130 GOSUB730:PRINT" EXPLORACION"
    :LOCATE,8
140 X=30:Y=6:GOSUB820
150 LOCATE6,10:PRINT"1 Leer cint
    a"
160 LOCATE6,12:PRINT"2 Buscar pr
    ograma"
170 LOCATE6,14:PRINT"3 Salida mo
    do exploración"
180 NO=3:GOSUB1050
190 IFOP=3THENOP=0:RETURN
200 N$="":IFOP=2THENGOSUB1150
210 CLS:PRINT"Apriete Play en ca
    ssete":PRINT"Para finalizar
    , [ctrl+stop]"
220 IFNP=100THENBEEP:PRINT"Memor
    ia llena.Para entrar otro p
    rograma borrar existentes."
    :MOTOROFF:GOSUB1110:RETURN
230 A$=USR(STRING$(20,0))
240 IFASC(A$)=0THENBEEP:GOTO230
250 T=ASC(A$)
260 IFT=&HDTHENMOTOR OFF:RETURN
270 NP=NP+1:TP(NP)=0
280 IFT=&HD0THENTP(NP)=1
290 IFT=&HEATHENTP(NP)=2
300 IFT=&HD3THENTP(NP)=3
310 PR$(NP)=MID$(A$,2,6)
320 IN(NP)=ASC(MID$(A$,8,1))+ASC
    (MID$(A$,9,1))*256
330 FI(NP)=ASC(MID$(A$,10,1))+AS
    C(MID$(A$,11,1))*256
340 EJ(NP)=ASC(MID$(A$,12,1))+AS
    C(MID$(A$,13,1))*256
350 PRINT"He encontrado: "PR$(NP
    )" Carga: "TP$(TP(NP))
360 IFNP=100GOTO220
370 IFN$<>PR$(NP)GOTO230
380 PRINT"Hallado":MOTOR OFF:GOS
    UB1110
390 RETURN
400 ' Información última cinta
410 GOSUB730:PRINT"INFORMACION U
    .C."
420 X=39:Y=10:GOSUB820
430 LOCATE1,10:PRINT" 1 Listado
    de programas encontrados"
440 LOCATE1,12:PRINT" 2 Informac
    ión programa"
450 LOCATE1,14:PRINT" 3 Búsqueda
     prog entre los hallados"
460 LOCATE1,16:PRINT" 4 Borrado
    datos"
470 LOCATE1,18:PRINT" 5 Salida m
    odo información"
480 NO=5:GOSUB1050
490 ONOPGOSUB520,580,870,940
500 IFOP-5GOTO400
510 RETURN
520 CLS:IFNP=0THENBEEP:PRINT"No
    hay programas en memoria":G
    OTO560
530 FORT=1TONP
540 PRINT"Programa número"T":"PR
    $(T)
550 NEXTT
560 GOSUB1110
570 RETURN
580 LOCATE26,12:PRINT"Número:__"
590 LOCATE33,12:P=VAL(INPUT$(1))
    :PRINTUSING"#";P
600 LOCATE34,12:P=P*10+VAL(INPUT
    $(1)):PRINTUSING"#";PMOD10
610 CLS:IFP=0ORP>NPTHENBEEP:PRIN
    T"Programa inexistente.":GO
    SUB1110:RETURN
620 GOSUB730:PRINT"PROGRAMA"P
630 LOCATE2,8:PRINT"Título: "PR$
    (P)" {";
640 FORT=1TO6:PRINTHEX$(ASC(MID$
    (PR$(P),T,1)))SPACE$(1);:NE
    XTT:PRINT"}"
650 LOCATE2,10:PRINT"Tipo de car
    ga:"TP$(TP(P))
660 IFTP(P)-1GOTO710
670 LOCATE2,12:PRINT"Direcciones
    :Inicio----"HEX$(IN(P))
680 LOCATE14:PRINT"Fin-------"HE
    X$(FI(P))
```

```
690 LOCATE14:PRINT"Ejecución-"HE
    X$(EJ(P))
700 PRINT:PRINT"  Memoria ocupad
    a:"FIX((FI(P)-IN(P))/1024)"
    Kbytes"
710 GOSUB1110
720 RETURN
730 ' Dibuja cabecera
740 CLS
750 PRINTSTRING$(39,42)
760 FORT=0TO2
770 PRINTCHR$(42)SPACE$(37)CHR$(
    42)
780 NEXTT
790 PRINTSTRING$(39,42)
800 LOCATE12,2
810 RETURN
820 'Dibuja cuadrado
830 LOCATE0,8:PRINTSPACE$((38-X)
    /2)STRING$(X,45)
840 FORT=0TOY:PRINTSPACE$((38-X)
    /2)CHR$(124)SPACE$(X-2)CHR$
    (124):NEXTT
850 PRINTSPACE$((38-X)/2)STRING$
    (X,45)
860 RETURN
870 GOSUB1150
880 LOCATE0,15:PRINT"No hallado"
    :LOCATE0,15
890 FORT=1TONP
900 IFPR$(T)=N$THENPRINT"Program
    a número:"T
910 NEXTT:PRINT
920 GOSUB1110
930 RETURN
940 LOCATE17,16:PRINT".¿Seguro?"
    ;
950 A$=INPUT$(1):IFA$="S"ORA$="s
    "THENNP=0
960 RETURN
970 ' Menú principal
980 GOSUB730
990 PRINT"MENU PRINCIPAL"
1000 X=30:Y=6:GOSUB820
1010 LOCATE5,10:PRINT" 1 Explora
     ción cinta"
1020 LOCATE5,12:PRINT" 2 Informa
     ción última cinta"
1030 LOCATE5,14:PRINT" 3 Fin"
1040 RETURN
1050 ' Selección de opción
1060 LOCATE8,22:PRINT"(seleccion
     e opción < >)"
1070 LOCATE28,22
1080 OP=VAL(INPUT$(1)):PRINTUSIN
     G"#";OP
1090 IFOP=0OROP>NOGOTO1070
1100 RETURN
1110 'Espera tecla
1120 LOCATE12,22:PRINT"Pulse una
      tecla"
1130 IFINKEY$=""GOTO1130
1140 RETURN
1150 ' Espera título
1160 GOSUB730:PRINT" BUSCADOR PR.
     "
1170 X=17:Y=2:GOSUB820
1180 LOCATE12,10:PRINT"Nombre: _
     _____":N$=""
1190 FORT=1TO6:LOCATE19+T,10
1200 A$=INPUT$(1)
1210 IFASC(A$)<32GOTO1200
1220 PRINTA$:N$=N$+A$:NEXT
1230 RETURN
1240 'Inicialización
1250 SCREEN0:WIDTH40
1260 PRINT"EXPLORADOR CINTAS MSX
     "
1270 PRINT"versión 1.0 J2C 1985"
1280 PRINT"cargando datos"
1290 DIMTP$(4),PR$(100),TP(100),
     IN(100),FI(100),EJ(100)
1300 TP$(1)="BLOAD":TP$(2)="LOAD
     ":TP$(3)="CLOAD"
1310 FORT=&HDA00TO&HDA44:READA$:
     POKET,VAL("&H"+A$):S=S+VAL(
     "&H"+A$):NEXT:DEFUSR=&HDA00
1320 IFS-8966THENPRINT"Error en
     datas":STOP
1330 KEY OFF:RETURN
1340 RETURN 70
1350 DATA13,1A,6F,13,1A,67,E5,CD
     ,E1,0,CD,E4,0,4F,6,9,C5,CD,
     E4,0,C1,B9,28,2,E1
1360 DATAC9,10,F4,FE,D0,28,C,FE,
     EA,28,8,FE,D,28,4,FE,D3,20,
     EC,E1,77,23,CD,37
1370 DATADA,E5,CD,E1,0,E1,6,6,E5
     ,C5,CD,E4,0,C1,E1,77,23,10,
     F5,C9
```

La memoria de vídeo (V)

Habiendo ya finalizado el estudio de los cuatro sistemas de almacenamiento de la memoria de vídeo, pasamos en este número a analizar las tablas referentes al tratamiento de las figuras móviles, los «sprites».

los sprites

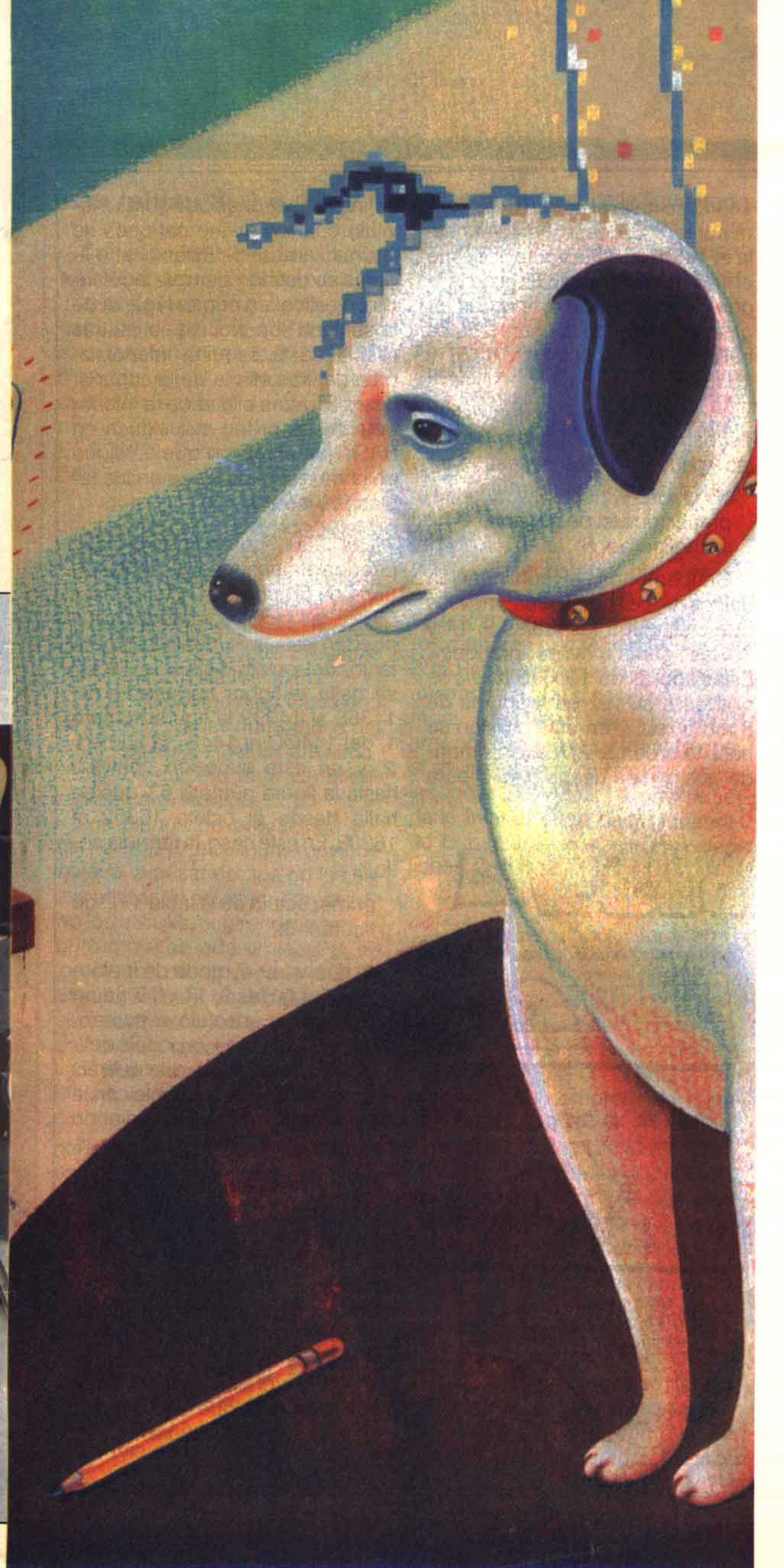


Los *sprites* MSX o figuras móviles, que ya fueron estudiadas en el número 5 de *MSX Magazine,* son figuras monocromáticas enmarcadas en cuadrados de 8 o 16 *píxels* de lado, o en cuadrados de 16 o 32 *píxels* de lado si se trata de figuras de *píxels* ampliados. Su mayor ventaja es la movilidad que tienen por la pantalla, pues son capaces de adquirir una gran velocidad, incluso desde *BASIC.*

Pero gracias a la memoria de vídeo, gran parte del *BASIC* ha quedado atrás, y podemos hablar directamente con la máquina, obteniendo una ejecución más rápida de nuestras órdenes. En este aspecto las figuras móviles también quedan incluidas, y va a ser posible definir *sprites,* moverlos y hacerlos desaparecer desde la memoria de vídeo, así como cambiarlos de color y de forma.

Hemos preferido dedicarnos al tema de los *sprites* tras haber concluido el estudio de los cuatro sistemas de almacenamiento, a causa de que las tablas que se ocupan de las figuras móviles en los sistemas de almacenamiento de los modos *SCREEN 1, SCREEN 2* y *SCREEN 3* consevan su manera especial de almacenar los datos estén en el modo que estén. De esta manera, cuando hablamos de la tabla 9, por ejemplo, estamos hablando también de la tabla 14 y de la tabla 19, pues las tres son una sola, que adopta diferentes números dependiendo del sistema de almacenamiento en el que se encuentre. Por otra parte, también cuando hablamos de la tabla 8 hablamos de las tablas 13 y 18, pues son a la vez una misma tabla. Las tablas 8 y 9 pertenecen al sistema de almacenamiento del modo *SCREEN 1,* la 13 y la 14 al del modo *SCREEN 2* y la 18 y la 19 del modo *SCREEN 3.*

A diferencia de los demás, el sistema de almacenamiento del modo *SCREEN Ø* carece de la posibilidad de las figuras móviles,



quizás porque los diseñadores de los ordenadores de la norma MSX pretendían concederle un carácter más sobrio, a causa de que todo ordenador ha de estar preparado para ejecutar todo tipo de programas, y para los programas mal llamados «programas serios» era necesario un modo de texto práctico, escueto, con gran cantidad de columnas (4Ø columnas, no como en el modo *SCREEN 1*, que tiene 32) y que no necesita figuras móviles. Sin embargo, para el resto de los programas sí vendría bien establecer figuras móviles, y para eso se debió crear el modo *SCREEN 1*.

Los modos de gráficos de alta definición y multicolor requerían forzosammente figuras móviles, y fueron establecidas sus tablas idénticamente a las tablas de figuras móviles del modo *SCREEN 1*, e incluso en los mismos octetos de la memoria de vídeo.

Las tablas 9, 14 y 19

Vamos a explicar en primer lugar el funcionamiento de las tablas 9, 14 y 19, pues son mucho más sencillas de manejar y más fáciles de comprender que las tablas 8, 13 y 18.

Las tablas 9, 14 y 19 están destinadas a retener los patrones de todas las figuras móviles que pueden aparecer en pantalla. Su longitud es de 2Ø48 octetos y en nuestros ordenadores MSX comienzan en el octeto 14336, para terminar en el octeto 16383, último octeto de la memoria de vídeo.

Como sabes, existen cuatro tipos de figuras móviles: de 8 x 8 *píxels* sin ampliar, de 8 x 8 *píxels* ampliados, de 16 x 16 sin ampiar y de 16 x 16 ampiados. Si el tipo de figuras móviles establecido es uno de los dos primeros, las tablas 9,

14 y 19 almacenan los datos de una manera, y si es de los dos últimos de otra.

El hecho de que los *píxels* estén ampliados o no, se especifica en otra parte de la memoria de vídeo, NO EN LAS TABLAS, sino en los registros *VDP* (Video Display Processor) de los que no hemos hablado hasta ahora, pero que trataremos en los números siguientes. Así, al no poder distinguir las tablas si los *píxels* son ampliados o no, sólo tratan dos tipos de figuras móviles: las «grandes» de 16 x 16, y las «pequeñas», de 8 x 8 *píxels.*

Aunque el hecho de que las figuras móviles sean «grandes» o «pequeñas» también se almacena en los registros *VDP,* esto sí afecta al funcionamiento de las tablas 9, 14 y 19. Si las figuras móviles son «pequeñas», los 2Ø48 octetos de cada tabla se dividen en 256 grupos de 8 octetos, numerados del Ø al 155. Cada grupo es un patrón de figura móvil, y funciona igual que los patrones de los caracteres: 8 filas x 8 columnas de *píxels* se corresponden con 8 octetos x 8 cifras en binario cada uno. Si la cifra en binario es un 1, ese *píxel* se impregna del color que le hayamos impuesto a la figura móvil, pero si la cifra en binario es un Ø, NO SE PINTA DEL COLOR DE FONDO, sino que simplemente no se impregna de ningún color, se queda transparente, y gracias a su transparencia permite que sea visible el color del *píxel* que se halla en el plano pricipal, detrás de todos los planos de proyección. Ese *píxel* puede tener color de tinta o color de fondo, pues la figura móvil que se encuentra en esa zona de la pantalla no influye para nada en ese asunto.

Los patrones de figura móvil se distribuyen del siguiente modo: desde el octeto 14336 hasta el 14343 se halla el patrón de la figura número Ø, desde el 14344 hasta el 14351 el de la figura 1, desde el 14352 hasta el 14359 el de la figura 2, y siguiendo este orden lógico, el patrón de la figura 255 se halla entre los octetos 16376 y 16383.

Para averiguar el primer octeto del patrón de una figura móvil de las «pequeñas», es necesario introducir la siguiente fórmula:

primer octeto de la tabla + n.º de la figura x 8

Ocupémonos a continuación del otro tipo de figuras móviles que distinguen las tablas 9, 14 y 19: las figuras móviles «grandes». Estas figuras de 16 x 16 *píxels* se dividen a su vez en 4 figuras de 8 x 8 *píxels,* y en esta división se basa la orden *SPRITE$* (el comando que se usa desde *BASIC* para definir los patrones de figuras móviles) cuando define patrones de figuras «grandes». Estas cuatro figuras se definen siempre siguiendo un orden: en primer lugar la de la esquina superior izquierda, tras ella la de la esquina inferior izquierda, tras ella la de la superior derecha y tras ella la de la inferior derecha. El orden que siguen en *BASIC* es el mismo que se sigue en la memoria de vídeo en las tablas 9, 14 y 19.

Las 256 figuras móviles que pueden almacenar estas tablas se unen en grupos de 4, y forman las 64 figuras móviles «grandes» que pueden salir a la pantalla. Por lo tanto, del octeto 14336 al 14367 se halla la figura número Ø, del 14368 al 14399 la figura número 1, del 144ØØ al 14431 la número 2, y en esta sucesión continúa hasta la figura número 63, que se halla desde el octeto 16352 al 16383. En este caso, la fórmula sería:

primer octeto de la tabla + n.º de figura x 32

Al encontrar el modo de funcionar de las tablas 9, 14 y 19, ya no necesitáis en absoluto el comando *"SPRITE$",* pues ya podéis definir las figuras móviles que queráis comunicando directamente con la memoria de vídeo y disponiendo su almacenamiento en las tablas.

Como ya hemos explicado, las tablas 9, 14 y 19 sirven para almacenar en memoria los patrones de las figuras móviles. Debido a ello, sus aplicaciones prácticas no podremos verlas en la pantalla hasta que no sepamos situar los patrones debidamente, con una posición y un color concreto, en la zona de la pantalla que deseemos, y eso se logra con las tablas 8, 13 y 18.

Las tablas 8, 3 y 18

Estas son las tablas destinadas a almacenar los datos de posición y color de las figuras móviles. Tienen una longitud de 128 octetos y en nuestros ordenadores MSX comienzan en el octeto 6912, en cualquier sistema de almacenamiento excepto en el del modo *SCREEN Ø*, en el que, como ya sabéis, no hay figuras móviles.

Suponemos que ya poseéis conocimientos acerca de los 32 planos de proyección de figuras móviles que existen en la pantalla (para ampliar estos conocimientos podéis consultar el artículo "SPRITES MSX: DIBUJOS ANIMADOS EN TU ORDENADORd" en el n.° 5 de MSX Magazine). Cada uno de esos 32 planos numerados del Ø al 31 posee 4 octetos para almacenar los datos que le caracterizan en cada instante.

Los planos de proyección tienen siempre una figura móvil dibujada, a pesar de que no los estéis utilizando en ese momento. Si no los utilizáis, contrendrán una figura móvil situada en la zona de los planos de proyección que se encuentra fuera de la pantalla. Por eso, aunque no los estéis utilizando, tendrán unos datos característicos que han de guardar en las tablas 8, 13 y 18 en cada instante.

Ya hemos dicho que cada plano dispone de 4 octetos, en cada uno de los cuales introduce un dato. Los datos a introducir son los siguientes:

— La coordenada «y» del *píxel* de aplicación del *sprite*. Esta coordenada se introduce en el primero de los 4 octetos de los que dispone un plano de proyección, (el punto de aplicación del *sprite* es el *píxel* que al situarse el *sprite* en la pantalla pasa a ser el vértice superior izquierdo del cuadrado en el que está enmarcada la figura móvil). Como ya sabéis, cada plano de proyección contiene 256 filas x 256 columnas de *píxels*, y al tener la pantalla únicamente 192 filas x 256 columnas, hay un espacio bajo la pantalla donde los *sprites* pueden encontrarse, pero no pueden ser vistos por el observador. Por lo tanto, si esta coordenada tiene un valor mayor o igual a 192, vuestra figura móvil no se verá, y si tiene un valor comprendido entre 184 y 191 se verá parcialmente. Por otra parte, en la memoria de vídeo, el fenómeno del *«wrap-around»* no se da, pues en un octeto es imposible introducir números menores que Ø o mayores de 255. Pero es necesario puntualizar aún más. Lo que sí se da en la memoria de vídeo es el *«wrap-around»* parcial, de tal manera que si situamos una figura móvil entre las líneas 249 y 255, será visible su parte inferior en la parte superior de la pantalla, es decir, entre las líneas Ø y 7.

— La coordenada «x» del *píxel* de aplicación del *sprite*. Se introduce en el segundo de los 4 octetos. En esta coordenada tampoco se da el *«wrap-around»*, pero ni siquiera se da el *«wrap-around»* parcial que sí se daba en la coordenada «y». Si la coordenada «x» se encuentra comprendida entre 248 y 255, no se verá más que la parte izquierda de la figura móvil a la derecha de la pantalla, y no será visible la otra parte a la izquierda de la pantalla, como sería lógico si existiera el *«wrap-around»*.

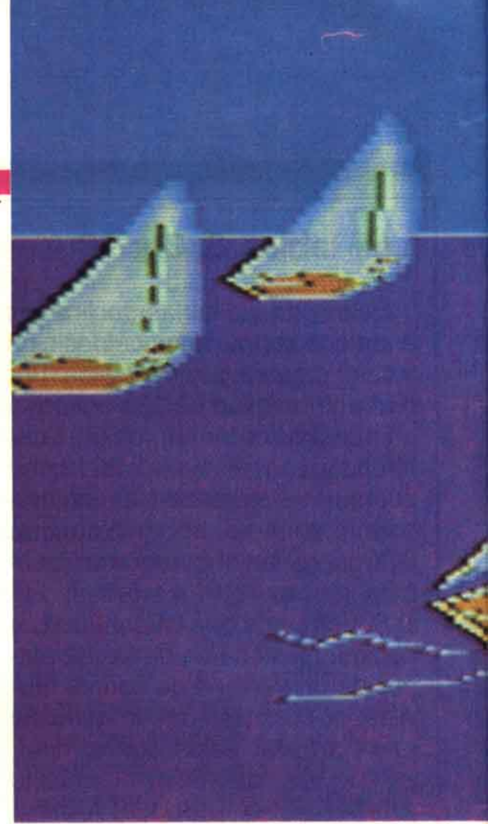
— El tercer dato es el número de figura móvil. Como ya dijimos antes, los patrones de las figuras móviles se almacenan en las tablas 9, 14 y 19, y en el caso de que sean figuras móviles «pequeñas» de 8 x 8 *píxels* caben un total de 256 patrones, numerados del Ø al 255. Este número de patrón es el que hay que introducir en el tercer octeto. Sin embargo, en el caso de que sean figuras móviles «grandes», sólo hay 64 patrones, que antes numerábamos del Ø al 63. Pero esta numeración no es válida para las tablas 8, 13 y 18, y hay que «renumerar» estos patrones, multiplicando por 4 su número anterior. De esa manera el patrón que antes era el Ø sigue siendo el Ø, el que era antes el 1 es ahora el 4, el que antes era el 2 es ahora el 8, y continuando esta sucesión, el que antes era el 63 es ahora el 252. Los números nuevos son los que debemos introducir en el tercer octeto. Pero si en vez de Ø, 4, 8, 12, etc. introducimos un número que no es múltiplo de 4 (el 15, por ejemplo), el resultado es el mismo que si hubiéramos introducido el múltiplo de 4 inmediatamente anterior (en el caso del 15, habría sido el 12).

— El cuarto dato es el color de la figura móvil. Será mejor tratar a este dato en hexadecimal, y entonces, al ser un número entre Ø y 255, tendrá 2 cifras hexadecimales. La segunda cifra hexadecimal es el color que tendrá el *sprite* que contiene ese plano de proyección (la lista con los colores y su código en hexadecimal se halla en la segunda parte de esta serie de artículos, en MSX Magazine n.° 8). La primera cifra, sin embargo, tiene una función absolutamente distinta a la de la segunda. Si la primera cifra hexadecimal toma un valor comprendido entre el Ø y el 7, no sucede nada, pero si toma un valor entre el 8 y el F, súbitamente la figura móvil se sitúa 32 *píxels* a la izquierda de donde se hallaba. Es decir, el efecto es el mismo que si restáramos 32 a la coordenada «x» de la figura móvil, aunque en realidad la coordenada «x» no ha variado en absoluto. Si la «x» es

menor de 24, y se cambia así la primera cifra, la figura móvil desaparece de la pantalla. Pero, ¿para qué sirve este hábil truco de la memoria de vídeo? Si os habéis fijado, para que una figura móvil sea sólo parcialmente visible hay que situarla o en el límite superior de la pantalla (dándole a la coordenada «y» valores del 249 al 255) o en el límite inferior (dándole valores del 184 al 191) o en el límite derecho (dándole, en este caso a la coordenada «x», valores de 248 a 255), pero en el límite izquierdo de la pantalla hasta ahora era imposible. Gracias a ese deplazamiento a la izquierda de la primera cifra en hexadecimal, si situáis la coordenada «x» entre 24 y 31 y la primera cifra hexadecimal del cuarto octeto es mayor o igual a 8, vuestra figura móvil aparecerá sólo en parte en el límite izquierdo de la pantalla. Aclaremos eso un poco más: si situamos un 8, por ejemplo, en la primera cifra hexadecimal, la figura móvil se desplaza 32 *píxels* a la izquierda, pero si se cambia a un 9, no se desplaza otros 32, sino que permanece inmóvil en su posición desplazada, así como en el caso de que situemos de nuevo un número menor que 8, la figura recupera su posición normal (x,y) según indican los dos primeros octetos.

Para explicar todo lo anterior, hemos partido de la base de que cada plano de proyección tiene 4 octetos, pero ¿cómo están distribuidos en la memoria de vídeo? Muy sencillo. Los octetos del 6912 al 6915 corresponden al plano Ø, los del 6916 al 6919 al plano 1, los del 692Ø al 6923 al plano 2, y siguiendo esta sucesión, los del 4Ø36 al 4Ø39 al plano 31.

La fórmula para hallar el octeto que se ocupa de la coordenada «y» de un plano de proyección es la siguiente:



primer octeto de la tabla + n.º de plano x 4

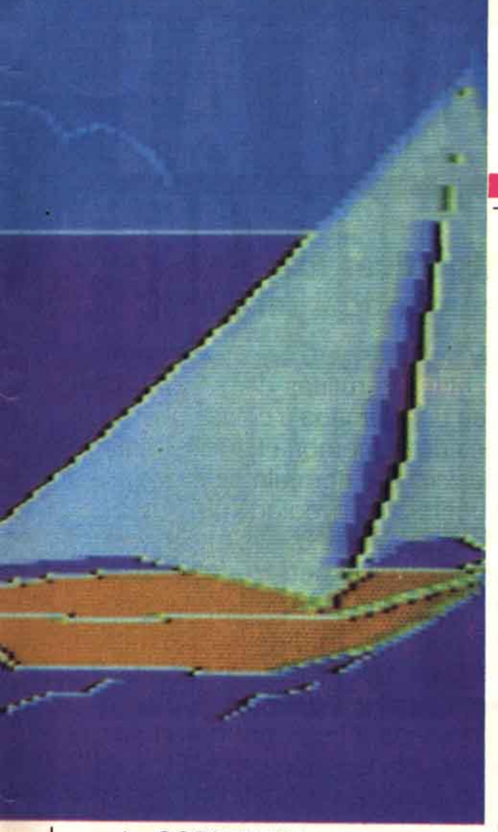
Para hallar el de la coordenada «x» habrá que sumarle 1, para hallar el de la figura móvil hay que sumarle 2 y para hallar el del color hay que sumarle 3.





Las figuras móviles en la práctica

Podríamos tomar como ejemplo cualquier patrón de figura móvil, pero hemos escogido esta vez unos patrones que podréis utilizar en vuestros programas, y son el resultado de la identificación de patrones de caracteres y patrones de figura móvil. Introduciendo patrones de caracteres en los patrones de figura móvil se pueden lograr efectos gráficos muy vistosos en vuestros programas. Esta identificación ha de hacerse forzosamente en el modo *SCREEN 1*, pues el único sistema de almacenamiento que dispone a la vez de una lista de patrones de caracteres (tabla 7) y de una tabla de patrones de figuras móviles (tabla 9) (también podría hacerse desde el



modo *SCREEN Ø,* pero para eso falta saber las transiciones entre las tablas, que serán estudiadas en el próximo número).

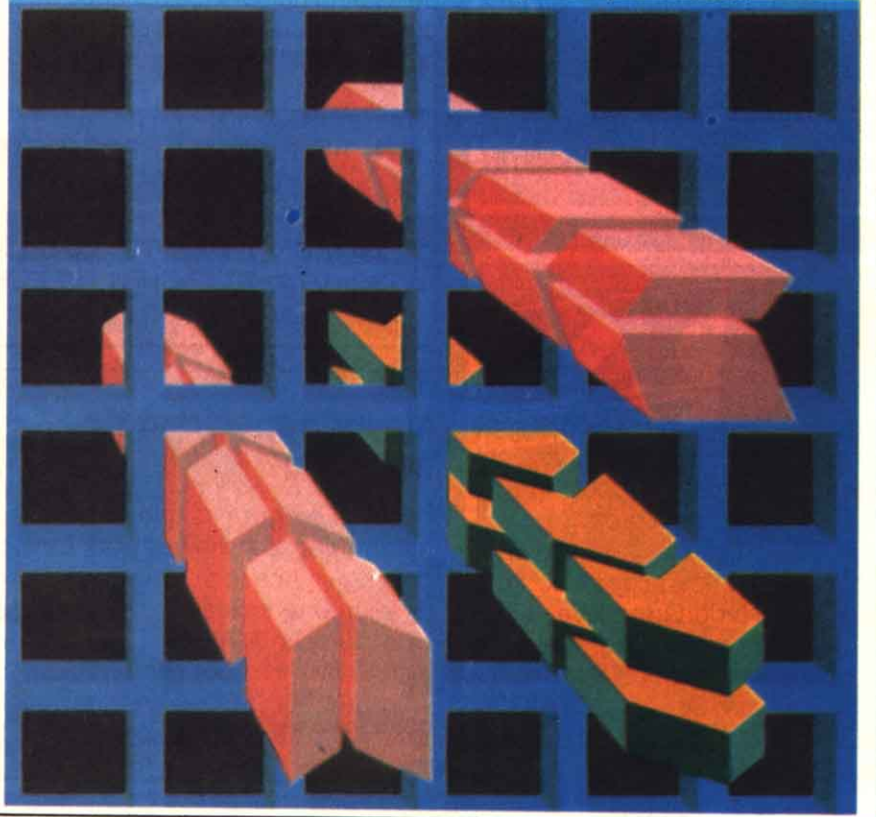
Pero vosotros os estaréis preguntando para qué hace falta esa identificación para lograr situar letras en la pantalla, cuando en un modo de texto como es el modo *SCREEN 1* las letras pueden ponerse en la pantalla con la orden *PRINT,* y la respuesta es inmediata. Con la orden *PRINT* sólo se puede lograr situar las letras en cuadros concretos de la pantalla, y si al querer moverlas lo que hacemos es ir pintándolas cuadro por cuadro (no hay otra manera) el observador ve demasiado bien que la letra está pegando «saltitos» de cuadro en cuadro. Sin embargo, si la convertimos en una figura móvil, además de tener mucha más libertad a la hora de impregnarla de un color concreto, se puede mover *píxel* a *píxel,* dando una total impresión de movimiento. Este programa es un pequeño ejemplo de cómo funciona esta identificación:

```
1Ø COLOR 4,15,15:SCREEN
   1:KEY OFF
2Ø DATA 77,83,88
3Ø FOR T=Ø TO 2:READ Q
4Ø FOR S=Ø TO 7:VPOKE
   14336 + T x 8 + S, VPEEK (Q x
   8 + S): NEXT:NEXT
5Ø FOR T=Ø TO 2:VPOKE 6914
   + T x 4, T:NEXT
6Ø FOR T=Ø TO 2:VPOKE 6915
   + T 4,8 + T x 2:NEXT
7Ø FOR T=Ø TO 2:VPOKE 6913
   + T 4,T x 8:NEXT
8Ø FOR T=Ø TO 2ØØ:FOR S=Ø
   TO 2:VPOKE 6912 + S x 4,T
9Ø VPOKE 6913 + S x 4, VPEEK
   (6913 + S x 4) + 1:NEXT:NEXT
```

Si acabáis de conectar vuestro ordenador MSX, y no habéis utilizado las figuras móviles desde entonces, podéis no copiar la orden 5Ø, pues sólo se ocupa de inicializar los números de figura móvil en función de los planos de proyección. El *DATA* de la orden 2Ø contiene los números de los caracteres «M», «S» y «X», que van a aparecer en pantalla mediante este programa. El bucle doble entre las órdenes 3Ø y 4Ø identifica los patrones de las figuras móviles Ø, 1 y 2 con esos caracteres, la orden 6Ø dispone que esos caracteres van a pintarse de rojo, amarillo oscuro y verde oscuro, la orden 7Ø sitúa las coordenadas «x» en las columnas de cuadros Ø, 1 y 2, situadas a la izquierda de la pantalla, y las órdenes 8Ø y 9Ø mueven estas figuras móviles en diagonal por la pantalla.

Las aplicaciones de las figuras móviles son muchísimas, como ya dijimos en el n.° 5 de MSX Magazine. Cualquier ejemplo de los que allí vimos en *BASIC* puede perfectamente ser tratado en la memoria de vídeo, mediante las tablas 8, 9, 13, 18 y 19.

En el próximo número hablaremos de las transiciones entre las tablas de la memoria de vídeo MSX y sus aplicaciones.





Mejorar la velocidad de los programas

Volvemos a la carga con más trucos para mejorar la velocidad de ejecución de los programas:

1. Evitar siempre que sea posible el uso de matrices, es preferible utilizar más variables simples.

2. En los juegos utilizar las teclas del cursor para controlarlos (las instrucciones para su lectura son más rápidas que la función INKEY$).

Anular la unidad de disco

Los usuarios de unidades de disco habrán observado que algunos programas no funcionan teniendo ésta conectada. Normalmente esto tiene fácil solucio: desconectar la unidad de disco. Pero cuando el ordenador lleva la unidad incorporada y no se puede desconectar, la solución consiste en hacer un CLEAR 200,62336: NEW con lo que la unidad de disco queda anulada.

Copias de la pantalla

Muchos usuarios se quejan de que los ordenadores MSX no disponen de una instrucción que permita realizar copias de pantalla a impresora. Para solventar este pequeño fallo presentamos una corta rutina que permite hacer una copia de una pantalla en modo texto (SCREEN 0) en la impresora. El programa 2 es una ampliación de este rutina para los usuarios de unidad de disco y les permite sacar el directorio por impresora.

Modificar los comandos del BASIC

Muchas rutinas de la ROM hacen una llamada a ciertas posiciones de la memoria RAM. En estas posiciones normalmente se encuentra un código de retorno (201) que hace que la rutina continúe correctamente, si modificamos alguna de estas posiciones con un código de salto a otra dirección podemos interceptar el funcionamiento de una rutina determinada. El siguiente programa intercepta la rutina de LIST y hace una llamada a la rutina de CLS por lo que al ejecutar el comando LIST se hará un limpiado de pantalla.

```
10 REM *** CLS
   EXTENDIDO ***
20 POKE &HFF89,195
30 POKE &HFF8A,&HC3
40 POKE &HFF8B,0
50 POKE &HFF8C,201
```

```
1 REM ** VOLCADO DE LA PANTALLA DE TEXTO
   A IMPRESORA **
10 DIR=0
20 FOR FI=1 TO 20:FOR CO=1 TO 40
30 DT$=DT$+CHR$(VPEEK(DIR)):DIR=DIR+1
40 NEXT CO:PRINT DT$;:DT$="":NEXT FI
```

Mayor capacidad de memoria

En los programas que no utilice ningún tipo de fichero, podemos conseguir una mayor capacidad de memoria tecleando la instrucción directa MAXFILES=0. Esta instrucción nos permite la utilización de 29082 bytes en lugar de los 28815 habituales en estos ordenadores.

Hay que resaltar, que la memoria se aprovecha bastante bien, pero si utilizamos el modo de pantalla gráfica habrá que tener en cuenta que no podremos escribir en ella puesto que, como los lectores habituales ya sabrán, hay que abrir un fichero para ello, cosa que debido a la función de esta instrucción es imposible.

LA MAS IMPORTANTE EDITORIAL DE REVISTAS DE INFORMATICA EN CASTELLANO

El periódico INFORMATICO

EL SEMANARIO PROFESIONAL POR EXCELENCIA

ORDENADOR POPULAR

LA REVISTA LIDER DE LOS MICROS



LA PRIMERA REVISTA EN CASTELLANO PARA IBM PC Y COMPATIBLES



LA REVISTA IMPRESCINDIBLE PARA LOS INTERESADOS EN EL STANDAR JAPONES

commodore Magazine

LA DE MAYOR DIFUSION PARA ORDENADORES COMMODORE



SINCLAIR AL ALCANCE DE TODOS

Todospectrum

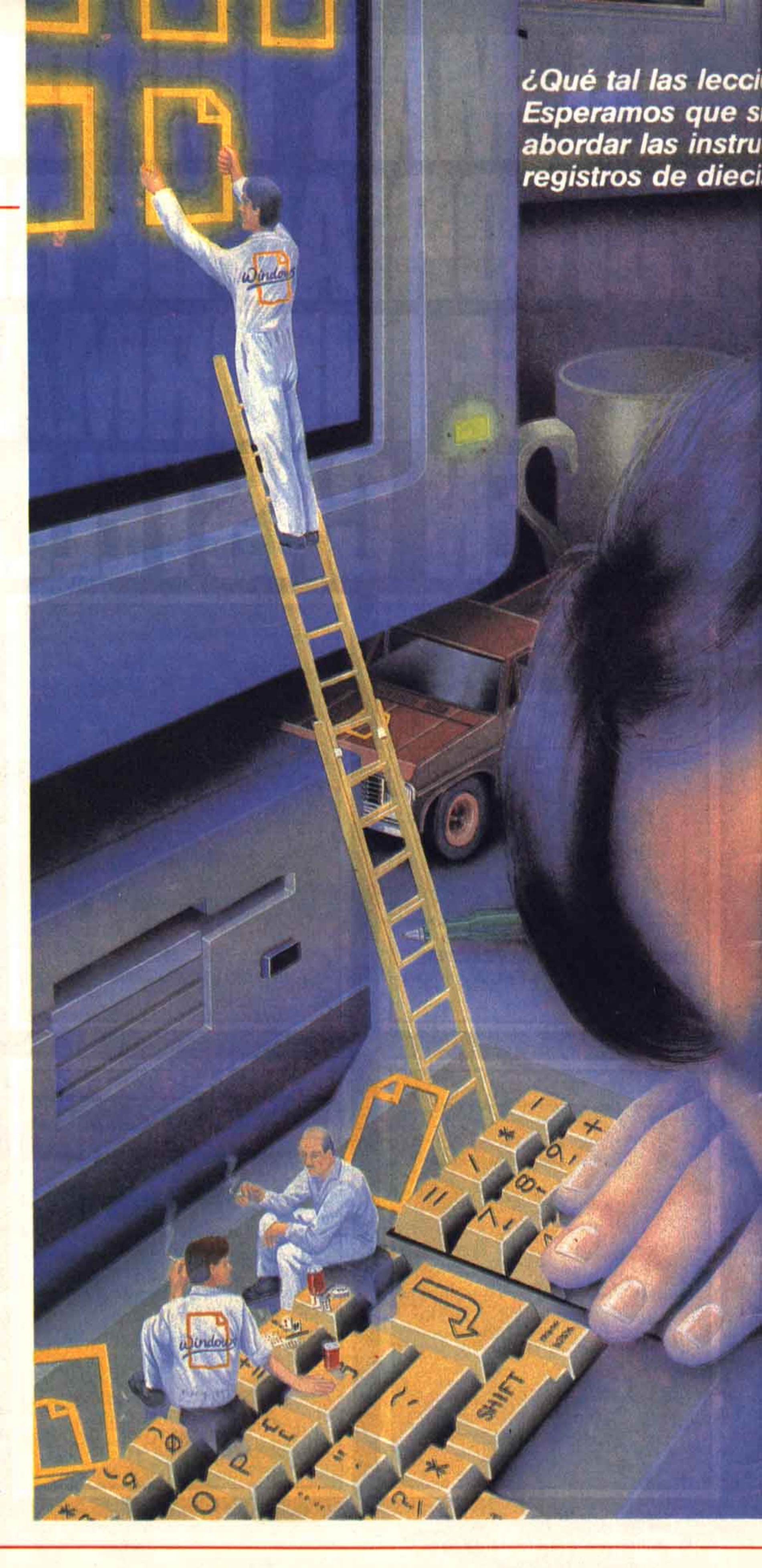
EL NIVEL MAS ALTO PARA SINCLAIR

publinformática, s/a

Bravo Murillo 377 - 28020 MADRID Tel (91) 733 74 13
Pelayo 12 - 08001 BARCELONA
Tels (93) 318 02 89 (93) 301 17 00 Ext 27 28

Como ya hemos comentado anteriormente, existen dos registros dobles en el *Z-80:* IX e IY. Además, algunos registros sencillos se pueden emparejar y funcionar como registros dobles. Estos son B y C (BC), D y E (DE), H y L (HL). Hay que hacer notar una diferencia muy importante entre el comportamiento como doble y como sencillo. Supongamos que tenemos los siguientes valores: B = 04H y C = FFH. Si ejecutamos la instrucción INC C, el registro C pasa a contener OOH pero B no resulta afectado, ya que el *bit* que nos llevamos (FF + 1 = 100H) se pierde, no pasa a B. Sin embargo, si ejecutamos INC BC, entonces C = 00H y B = 05H ya que la operación ya no es FFH + 1, si no 04FFH + 1.

También es necesario explicar lo que es el *stack.* Se trata de un área de memoria que se reserva para datos, y que puede ser manejada de diversas formas. Cuando se ejecuta una instrucción *CALL* (llamada a subrutina), la dirección siguiente a dicha instrucción queda almacenada en el *stack,* y es recuperada por la instrucción *RET* al final de la subrutina. Por otro lado, las instrucciones *PUSH* y *POP* también utilizan el *stack,* como ahora veremos. El control del *stack* o pila depende de un registro doble (SP) que se utiliza como puntero, es decir, que si queremos guardar un dato en dicha pila, la instrucción a ejecutar sería LD (SP), dato (¡OJO! esta instrucción no existe. La expongo para que comprendáis la idea). Además el puntero del *stack* se autoactualiza. Como las operaciones con la pila siempre implican el movimiento de dos *bytes,* el puntero se incrementa o decrementa en dos. También es importante se-



...nes anteriores? ¿Bien? En esta ocasión vamos a ...ciones que actúan sobre ...éis bits (registros dobles).

Código Máquina





ñalar que la pila crece hacia abajo. si el puntero indica, por ejemplo, BC06H, al guardar un dato éste se almacena en BC05H y BC04H, y el puntero pasa a señalar BC04H.

Hecha esta apreciación, comenzamos con el análisis instrucción por instrucción, empezando por el grupo de carga:

LD dd,nn.— dd representa uno de estos registros: BCc, DE, HL o SP. nn representa un número de dieciséis *bits* (dos *bytes*). Por ejemplo, LD DE,3A1FH hace que D contenga 3AH y E contenga 1FH.

LD IX,nn.— igual que el anterior, pero cargando el número en el registro doble IX.

LD IY,nn.— igual que el anterior, pero cargando el número en el registro doble IY.

LD HL,(nn).— en esta ocasión nn representa un número de dieciséis *bits* QUE APUNTA a la dirección de memoria en que se almacenan los dos *bytes* a guardar en HL. Veamos un ejemplo:

En la dirección 7F00H tenemos 16H.

En la dirección 7F01H tenemos BCH.

Si ejecutamos LD HL,(7F00H), L se carga con 16H y H con BCH. Atención, que no nos hemos equivocado. Primero se carga el *byte* bajo y luego el alto.

Esta instrucción ocupa tres *bytes,* uno para el código (21H) y dos para la dirección.

LD dd,(nn).— dd representa a BC, DE, HL o SP. Es idéntica a la anterior, si bien es más general y ocupa cuatro *bytes,* dos para el código y dos para la dirección.

LD IX,(nn).— igual a la anterior, pero con el registro IX.

LD IY,(nn).— igual a la anterior, pero con el registro IY.

LD (nn),HL.— es la «inversa» de LD HL,(nn), es decir, que en este caso el contenido de los registros se almacena en la dirección a la que apunta el número nn y la siguiente.

Veamos un ejemplo:

HL contiene C000H

Si ejecutamos LD (0000H),HL la dirección 0000H pasa a contener lo mismo que L (00H) y la dirección 0001H lo mismo que L (C0H). El registro HL no se altera, es decir, sigue conteniendo C000H después de ejecutar la instrucción.

LD (nn),dd.— igual que la anterior pero con BC, DE, HL o SP. Mientras que la anterior ocupa tres *bytes,* ésta ocupa cuatro.

LD (nn),IX.— igual que la anterior pero con el registro IX.

LD (nn),IY.— igual que la anterior pero con el registro IY.

LD SP,HL.— el contenido de HL pasa al registro SP. HL no se altera.

LD SP,IX.— el contenido de IX pasa al registro SP. IX no se altera.

LD SP,IY.— el contenido de IY pasa al registro SP. IY no se altera.

PUSH qq.— qq representa a uno de los registros BC, DE, HL o AF. Atención, que A y F no forman un registro doble. Simplemente, cuando hay que preservar uno de ellos, se preserva junto al otro, pero no existen operaciones tales como INC AF o ADD HL,AF ni nada parecido.

Esta instrucción envía el *byte* alto del registro correspondiente a (SP)−1 y el *byte* bajo a (SP)−2. Es decir, el proceso realizado es:

1. Decrementa el puntero del *stack* (SP = SP − 1)
2. Guarda el *byte* alto (LD (SP),qq − alto)
3. Decrementa el puntero del *stack* (SP = SP − 1)
4. Guarda el *byte* bajo (LD (SP),qq − bajo)

Como véis, en esta ocasión se almacena primero el *byte* alto y luego el bajo. El contenido del registro no queda alterado.

PUSH IX.— igual a la anterior, pero con el registro IX.

PUSH IY.— igual a la anteior, pero con el registro IY.

POP qq.— qq representa lo mismo que en PUSH qq. En esta ocasión recuperamos del *stack* el contenido del registro. El proceso es el siguiente:

1. Carga el *byte* bajo con el contenido de la posición de memoria a la que apunta el puntero del *stack*. (LD qq − bajo,(SP)).
2. Incrementa el puntero (SP= SP + 1).
3. Carga el *byte* alto (LD qq − alto,(SP)).
4. Incrementa el puntero (SP= SP + 1).

POP IX.— igual que la anterior, pero con el registro IX.

POP IY.— igual que la anterior, pero con el registro IY.

La instrucción *POP* no está obli-



gada a recuperar el mismo registro que salvó; es decir, no es obligado que después de PUSH HL, la primera instrucción *POP* que aparezca sea POP HL. Es perfectamente válida esta secuencia:

```
PUSH DE
POP HL
```

cuyo efecto es pasar el contenido del registro DE al HL sin que el primero quede afectado.

Ninguna instrucción de carga de dieciséis *bits* afecta a ningún *flag*.

Vamos ahora con las instrucciones aritméticas de dieciséis *bits*.

ADD HL,ss.— ss representa a BC, DE, HL o SP. Esta instrucción afecta al *flag* de acarreo, siempre que se produzca tal acarreo del decimosexto al decimoséptimo *bit*. El contenido de ss no resulta afectado. La suma se guarda en HL.

ADC HL,ss.— ss representa lo mismo que en el caso anterior. En esta ocasión, a la suma de HL y ss se le añade el acarreo existente con anterioridad. Esta instrucción afecta al *flag* de acarreo, al de cero, al de sobrepasamiento y al de signo (C, Z, V y S). Al igual que en



la anterior instrucción, el *flag* N se pone a cero, ya que la operación no es una resta.

SBC HL,ss.— ss representa lo mismo que en el caso anterior. La operación efectuada es HL-ss-carry, y afecta a los mismos *flags* que la anterior. El *flag* N se pone a uno, ya que la operación es una resta.

ADD IX,pp.— pp representa a BC, DE, IX o SP. Esta suma no añade el acarreo anterior, y sólo afecta

al *flag* de acarreo y al N, poniéndolo a cero.

ADD IY,rr.– rr representa a BC, DE, IY o SP. Posee las mismas características que la instrucción anterior.

INC ss.– ss representa a BC, DE, HL o SP. Su efecto es sumar uno a ss (ss = ss + 1). No afecta a ningún *flag*.

INC IX.– igual que la anterior,



pero con el registro IX.

INC IY.– igual que la anteior, pero con el registro IY.

DEC ss.– ss representa a BC, De, HL o SP. Su efecto es restar uno a ss (ss = ss – 1). No afecta a ningún *flag*.

DEC IX.– igual que la anterior, pero con el registro IX.

DEC IY.– igual que la anterior, pero con el registro IY.

Para finalizar este mes, vamos a ver las instrucciones de llamada y subrutina y retorno *(CALL* y *RET)*.

El concepto básico consiste en interrumpir en un momento dado el desarrollo del programa para ejecutar un subprograma y, una vez ejecutado éste, volver al programa anterior en la instrucción siguiente a la llamada al sub-programa. La utilidad de este sistema radica en que una tarea que puede ser necesaria varias veces sólo tenemos que escribirla una vez, y cuando la necesitemos nos bastará con escribir la instrucción *CALL* con la dirección en que empieza la rutina. Por ejemplo, podemos tener una subrutina que coja el número contenido en el registro A, lo interprete como código. *ASCII* y lo imprima en la pantalla. De este modo, si en un momento del programa queremos imprimir la letra «A», bastará con: LD a,41H; código ASCII de la A.

CALL PRINT; PRINT = dirección de la rutina de impresión. En vuestro MSX debéis escribir *CALL 00A2H*. Si no existiera la instrucción *CALL* tendríamos que escribir toda la rutina cada vez que la necesitáramos.

CALL nn.– nn representa un número de dieciséis *bits*. La dirección siguiente a la instrucción *CALL* se guarda en el *stack*, y la dirección nn se pasa al contador de programa PC.

CALL cc,nn.– cc es una condición. Si tal condición se cumple, se efectúa la instrucción *CALL* y por tanto la llamada a la subrutina. Si no se cumple la condición, se pasa a la siguiente instrucción. cc puede ser NZ, Z, NC, C, PO, PE, P, M.

RET.– esta instrucción debe situarse al final de una subrutina para poder regresar al programa desde el cual se la llamó. Esta instrucción recupera del *stack* los dos *bytes* que envió la instrucción *CALL* y los pasa al contador de Programa.

RET cc.– igual que la anterior, pero el hecho de que se efectúe el RETorno depende de la condición cc, que puede ser lo mismo que en el caso de CALL cc,nn.

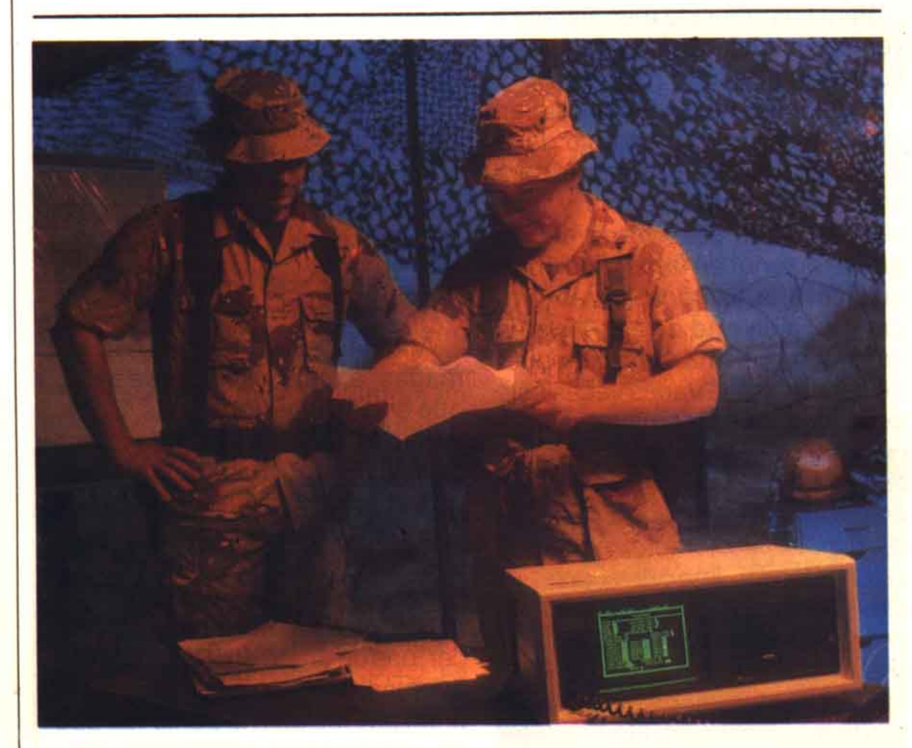
Además, existe otro tipo de llamadas a subrutinas especiales, que son las interrupciones. Como ya vimos, cuando se produce una interrupción, y según en qué modo trabaje el ordenador, se salta a una rutina que realiza una tarea concreta, como por ejemplo actualizar la pantalla o leer el teclado. Al final de esta rutina ha de aparecer una instrucción *RETI*, la cual, además de retornar al programa principal, restaura al periférico que generó la interrupción. Si la interrupción era No enmascarable, debe usarse *RETN*. Esta instrucción, además de retornar al programa principal, restaura el estado de las interrupciones enmascarables.

Por último, existen unas llamadas a subrutinas abreviadas, que ya adelantamos en el capítulo de las interrupciones: las instrucciones *RST*. En estas instrucciones la dirección va implícita, por lo que sólo ocupan un *byte*. Las direcciones llamadas pueden ser:

0000H
0008H
0010H
0018H
0020H
0028H
0030H
0038H

La forma general es *RST* p, donde p puede ser 08H, 10H, 18H, 20H, 28H, 30H o 38H.

Vamos a ver un ejemplo de có-









mo utilizar algunas de las instrucciones vistas hasta ahora. El propósito del ejercicio será imprimir en la pantalla un mensaje, por ejemplo un mensaje de error. Elegiremos uno muy familiar: Error de sintaxis.

Para imprimir el texto utilizaremos la rutina antes apuntada 00A2H, que imprime el carácter contenido en el registro A.

Analicemos:

Lo primero que hacemos es cargar el registro HL con la dirección en que comienza el texto. De este modo, utilizamos HL como puntero.

En LD A,(HL) comienza un bucle que repetiremos hasta que el acumulador sea cero. Cargamos una letra en A y en la instrucción siguiente la comparamos con cero. Si A es cero, después de la comparación el *flag* Z estará activado, y la instrucción *RET Z* nos hará volver al programa principal (en nuestro caso, el *BASIC*). Si A no es cero, el programa sigue su curso, imprimiendo el caracter correspondiente al código contenido en A. A continuación actualiza el puntero HL, para que señale al siguiente carácter, y vuelve al principio del bucle (LD A, (HL)) hasta que se encuentre con cero. Hemos añadido dos números detrás del texto. OAH ese el código del avance de línea, y 0DH es el código del retorno de carro. El primero hace que el cursor pase a la línea inferior, y el segundo lo envía a la columna 1 de la pantalla. Juntos producen el mismo efecto que si se pulsa la tecla *RETURN*.

Para cargar la rutina en memoria podéis utilizar este programa *BASIC:*

Una vez ejecutado, bastará con teclear A = USR(0) para que aparezca el mensaje en pantalla.

Intercambio ideas, programas, libros... Interesados en el código máquina, en la realización de programas y en la formación de un club. Escribir a Andrés Blanco Baulo, Apdo. 2168. Sabadell (Barcelona).

¿Quieres aprovechar toda la RAM de tu ordenador? Además intercambiamos programas, programamos y tenemos todo lo referente a tu MSX. Si quieres más información, escríbenos. Somos el Club L.S.D. Printer. Apartado 2093 de Murcia.

Intercambio programas MSX. Contactar con Daniel Delgado Segura. C/ Tomás Pérez Ubeda, 15. Casas-Ibáñez (Albacete).

Desearía contactar con personas que dispongan de modem para intercambiar información y realizar pruebas de transmisión de datos por teléfono. Mi ordenador es el SVI-738 X'-PRESS, con programas de comunicación en MSX y en CP/M 2.2. Los interesados contactar por carta y mandar su número de teléfono para hablar personalmente antes de realizar las pruebas. Mi modem es un Bondwell-101 de 300 baudios (V21). Dirigirse a Manuel López Cuesta. C/ Puentalarra, 18. 28031 Madrid.

Vendo ordenador Sony Hit-Bit 64K por 35.000 ptas. Nuevo a estrenar. Llamar al Tel.: (93) 217 99 74 de Barcelona de 11 a 15 horas ó 21 a 24 horas.

Vendo HB-75P + cassette de Sanyo + 5 cartuchos + 15 cassettes de juegos + cartucho data-cartridge + 28 revistas MSX + compilador de Pascal + 5 cintas vírgenes, todo por 70.000 ptas., en garantía y con facilidades de pago. Ponerse en contacto con Marti Llorach Torres. C/ Roger de Flor, 284. 08025 Barcelona.

Vendo Spectravídeo 728, con unidad de discos 707. Ha sido comprado en septiembre del 85. Todo por 90.000 ptas. Interesados llamar por las noches a Javier al Tel.: (91) 468 08 37.

Vendo unidad de disco Sony de 3.5 pulgadas, compatible con cualquier MSX y dos discos con programas. Todo por 52.000 ptas. Escribir a Antonio Marín, C/ Garita, 19. 07015 Palma de Mallorca, o llamar al Tel.: (971) 40 36 59.

Vendo Spectravídeo 328, con más de 100 personas de todo tipo, magnetófono, revistas y manuales. Interesados llamar al Tel.: (93) 335 17 26.

Deseo intercambiar programas para MSX. Muchos títulos. Escribir a Jesús López Alvarez. C/ Simancas, 2, 1.° B. Melilla, o llamar al Tel.: (952) 68 77 38.

Deseo vender un ordenador VG-8010, nuevo por 30.000 ptas. Escribir a Antonio Vãzquez Torre. C/ Góngora, 11. 14001 Córdoba.

Cambio cintas de juegos y compro cartuchos ROM. Contactar con Pablo Revuelta, C/ Artes Gráficas, 5. Valencia, o llamar al Tel.: 361 93 88.

Vendo Sony Hit-Bit 75P, 80K RAM, poco usado, con garantía por 3 meses, cables de conexión, manuales en español, 45 programas comerciales, por 45.000 ptas. Escribir a Francisco Angel Molina Montoro. Avda. de Barcelona, 326, 4.° B. 18006 Granada. O llamar al Tel.: (958) 11 96 86 (de 21 a 23 horas).



Rincón del lector

COMPATIBILIDAD ENTRE UNIDADES DE DISCO

Sé que todos los ordenadores MSX son compatibles entre sí, pero la pregunta es ¿y las unidades de disco? Ya no me refiero a las diferencias de tamaño, sino entre las del mismo tamaño. ¿Utilizan todos los MSX el mismo MSX-DOS? ¿Podemos leer los discos grabados con una unidad Sony en otra de la marca Spectravídeo (X-press), Philips, Canon o cualquier otra que tengan discos de 3.5 pulgadas?

J. M. Rial
Alicante

Hasta la fecha no hemos tenido problema alguno con las unidades de disco de 3.5 pulgadas, puesto que en la redacción estamos trabajando con el X'press, la unidad de Sony y la de Philips, sin haber tenido pega alguna. Ahora bien, los discos formateados a doble cara doble densidad no pueden ser leídos por una unidad de simple cara doble densidad, sin embargo lo contrario si puede ser posible. De cualquier manera, estamos trabajando en un artículo sobre las unidades de disco que publicaremos más adelante.

ERRORES EN UN PROGRAMA

El motivo de mi carta de doble. Por una parte quisiera sugeriros la creación de un apartado en vuestra publicación con la finalidad de dar a conocer todas las posibilidades de las diferentes instrucciones del BASIC MSX a las que los principiantes difícilmente podemos acceder con nuestros manuales.

Por otra parte, quisiera comentaros que después de haber escrito el programa «Despertador Musical» (pág. 60 del número de febrero), no hay forma de hacerlo funcionar, ya que se interrumpe nada más empezar y en la pantalla aparece el mensaje "Illegal function call in 60".

He repasado varias veces el programa y creo que no he cometido ningún error al pasarlo.

Santi Martínez i Simon
Carcelona

La serie sobre el BASIC, que pusimos en marcha hace unos meses, está dedicada a todos aquellos, bien sean expertos o principiantes, para aprovechar como bien dices, las posibilidades del estándar.

En cuanto a la segunda cuestión, el error que se produce se debe a que la función CHR$(D%) encuentra que el valor de D% no es adecuado. Como puede ver, D% se calcula en la línea 50 a partir de D%, y D% se obtiene de las líneas DATA (80-100) también en la línea 50 con READ D%. Por tanto, lógicamente el error debe estar en los datos de las líneas 80 a 100. A propósito, como en la revista no queda del todo claro, en estas líneas DATA lo que separa los númeeros son comas, no puntos. El error más frecuente es que hayas escrito una letra O en lugar de un número cero (0), pero puede ser otro. Revisa esas líneas.

TECLADO MUSICAL PARA TOSHIBA

He visto que para el Yamaha ha aparecido una especie de interface a través del cual puede ser conectado un teclado musical a este ordenador. No he podido conseguir ningún tipo de información sobre él, y aunque todos los MSX son compatibles (poseo Toshiba), quisiera saber si también podré acoplarlo a mi ordenador, para qué sirve dicho teclado y si existe algún tipo de software para convertir el ordenador MSX en un sintetizador como los que ya existen para otros ordenadores que tienen un generador de sonidos.

José Miguel Ros Vázquez
Barcelona

Efectivamente, Yamaha posee un teclado que se puede conectar al ordenador convirtiéndolo en un auténtico sintetizador musical. Sin embargo, Toshiba, también tiene dentro de su gama de periféricos, un teclado musical especialmente diseñado para ese ordenador. Por el momento, no estamos seguros que el de Yamaha se pueda conectar al Toshiba (ni a ningún otro), debido a que aquel está especialmente preparado para tal fin, ya que además de ser ordenador está orientado a ese sector de público cuyo «hobby» no sólo son los ordenadores, sinoque también lo complementan con la música. De todos modos, si deseas más información sobre cualquier periférico de Toshiba no dudes en dirigirte a:
E.M.S.A.
C/ Caballero, 79
08014 Barcelona

¿DONDE CONSEGUIR EL MSX-DOS?

El motivo de mi carta es el siguiente: poseo un ordenador Sony HB-75P, con unidad de discos Philips e impresora, con los cuales realizo diversas aplicaciones prácticas, no obstante me sería útil poder conseguir el disco con el sistema operativo MSX-DOS, pero hasta el momento ha sido totalmente imposible, por lo que les rogaría me comunicasen dónde podría conseguirlo.

Inocencio Piqueras Montes
Madrid

En teoría, las tiendas especializadas deberían tener ese sistema operativo y más cuando es tan importante y popular, junto con el CP/M. Caso de no ser así, te recomendamos dirigiros a los fabricantes directamente, ellos te resolverán la papeleta.

MSX

DYNADATA

NO LO DUDE



PRECIO INSUPERABLE

CONJUNTO:
ORDENADOR
MONITOR Y
CASSETTE
DYNADATA MSX

62.500

TARJETA
PROGRAMA.
EL NUEVO FORMATO DEL FUTURO.

MONITOR 12″
FOSFORO
VERDE

ADAPTADOR TARJETA

DATA CASSETTE

NUEVO

DYNADATA MSX
CON TECLADO EN
ESPAÑOL*

* Letras y signos iguales al teclado del PC de IBM.

Con la compra de este conjunto o del ordenador lleve gratis un joystick y tres cintas de divertidos juegos.

Ordenador DYNADATA MSX con teclado en español	39.900
DYNADATA MSX con cassette y monitor de color	95.500
DYNADATA MSX y unidad de diskette de 5 ¼″ de 360 Kbytes con monitor de fósforo verde	108.900
con monitor de color	141.000
TARJETA/PROGRAMA del juego LE MANS con adaptador, el cual sirve para cualquier tarjeta que usted adquiera. Precio especial con la compra del DYNADATA MSX	4.900

Los precios no incluyen I.V.A.

Con el DYNADATA MSX usted podrá:

- Divertirse con la amplia gama de juegos MSX.
- Aprender Informática y Basic con el curso autodidáctico y audiovisual.
- Llevar gestiones administrativas con los programas de proceso de textos, base de datos, contabilidad, stock, recibos, etc.
- Ayudar a sus hijos en sus estudios de 3.º a 8.º de EGB con los programas de Matemáticas, Lenguaje y Ciencias Naturales.
- Aprender idiomas tan necesarios como el inglés.
- Programar con los lenguajes: LOGO, PASCAL, FORTRAN y COBOL.

CURSO DE INFORMATICA Y BASIC

- Autodidáctico
- Audiovisual
- 12 cassettes
- 24 lecciones
- Evaluaciones periódicas
- Diploma Fin de Curso

INFORMATICA BASIC

DYNADATA MSX Y CURSO 61.900

Especificaciones DYNADATA MSX: Procesador Z80A, 64 Kbyte RAM, 16 Kbyte VRAM, 32 Kbyte ROM, 24 líneas x 40 columnas, 256 x 192 pixels, 16 colores, MSX-BASIC, MSX-DOS.

DYNADATA

Y:

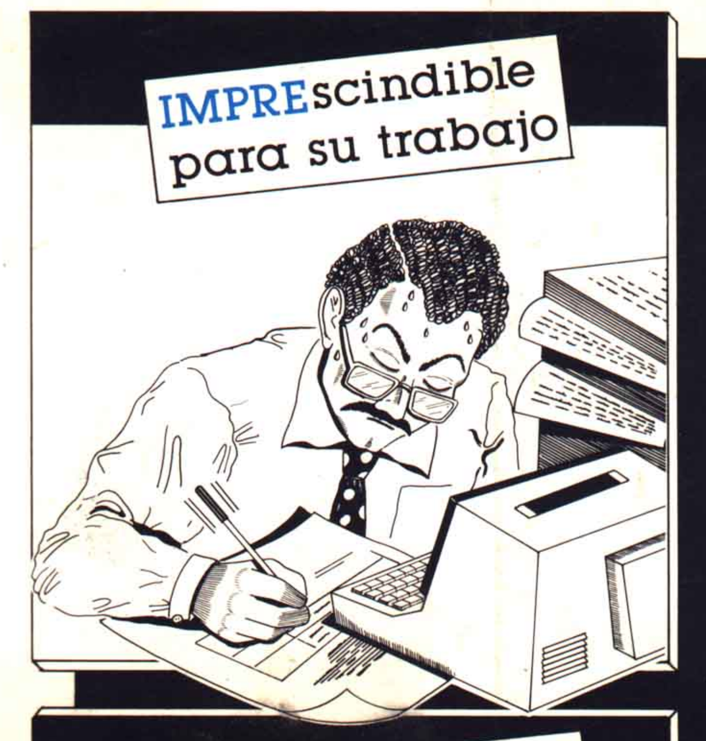
SONY
PHILIPS
CANON
SANYO
JVC
TOSHIBA
SPECTRAVIDEO
PIONEER
YAMAHA
MITSUBISHI
GOLDSTAR
SAMSUNG
HITACHI
MATSUSHITA
CASIO

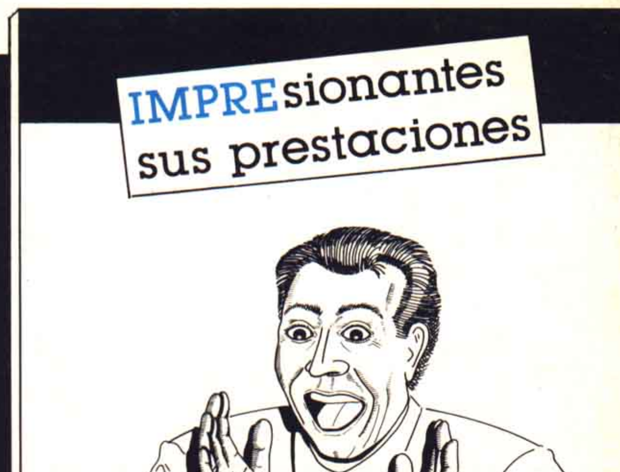
Se han decidido por MSX. Esto le permite compartir los programas y periféricos con todas estas reconocidas marcas.

Por todo, NO LO DUDE. Decídase por

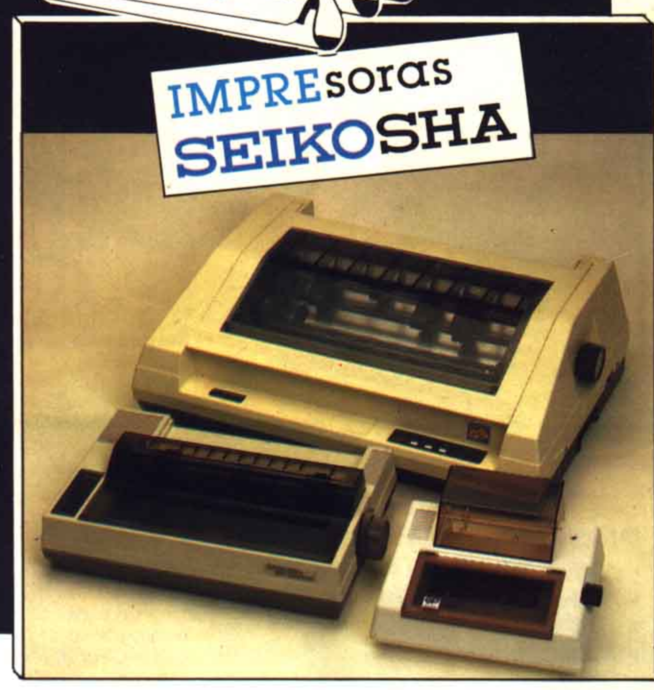


Solicite información: Sor Angela de la Cruz, 24 - 28020 Madrid. Tels. (91) 279 21 85 - 279 28 01 - 270 01 93. Telex 44619 DYNA
Delegación Barcelona: Aribau, 61, entlo - 08011 Barcelona. Tels. (93) 254 73 04 - 254 73 03









trebd tel.(96)362 42 15

GP-50 •	La pequeña 40 cps. Papel normal con interface paralelo, serial y Spectrum	17.990 ptas.
GP-700 *	La de color 50 cps. 7 colores. 80 columnas. Tracción y fricción. Papel de 10 pulgadas	64.990 ptas.
SP-1.000 *	La programable 100 cps.24 cps en alta calidad 96 cart. programables en RAM. Introductor hoja a hoja.♦	64.990 ptas.
SP-1.000AS	La programable 100 cps.24 cps en alta calidad con interface RS-232. Introductor hoja a hoja.♦	59.900 ptas.
MP-1.300AI	La polivalente 300 cps, 60 cps en alta calidad, interface paralelo y RS-232. Introductor hoja a hoja.♦&	119.900 ptas.
BP-5.200 *	La de oficina 200 cps, 106 en alta calidad.Buffer 4K.Carro de 15".Tracción y fricción.♦	199.900 ptas.
BP-5.420 *	La más rápida 420 cps. 106 cps en alta calidad. Buffer de 18K. Paralelo y RS-232.♦	339.900 ptas.

Interfaces: Serie RS-232C, Spectrum, IBM, COMMODORE, MSX, QL, Apple Macintosh, HP-IB

♦ Introductor automático de documentos opcional.

& Kit de color opcional.

* con interface paralelo

• con interface Spectrum

Nota: I.V.A. 12%, no incluido en los precios arriba indicados

Avda. Blasco Ibáñez, 116
Tel. (96) 372. 88. 89
Telex 62220 - 46022 VALENCIA

Muntaner, 60-2.º-4.ª
Tel. (93) 323. 32. 19
08011 BARCELONA

Agustin de Foxá, 25-3.º-A
Tels. (91) 733. 57. 00 - 733. 56. 50
28036 MADRID

